Dynamical Models
and
Machine Learning
for
Supervised Segmentation

Tony Shepherd

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
of the
University of London.

Department of Computer Science
University College London

September 15, 2009

To Anna

Abstract

This thesis is concerned with the problem of how to outline regions of interest in medical images, when the boundaries are weak or ambiguous and the region shapes are irregular. The focus on machine learning and interactivity leads to a common theme of the need to balance conflicting requirements. First, any machine learning method must strike a balance between how much it can learn and how well it generalises. Second, interactive methods must balance minimal user demand with maximal user control.

To address the problem of weak boundaries, methods of supervised texture classification are investigated that do not use explicit texture features. These methods enable prior knowledge about the image to benefit any segmentation framework. A chosen dynamic contour model, based on probabilistic boundary tracking, combines these image priors with efficient modes of interaction. We show the benefits of the texture classifiers over intensity and gradient-based image models, in both classification and boundary extraction.

To address the problem of irregular region shape, we devise a new type of statistical shape model (SSM) that does not use explicit boundary features or assume high-level similarity between region shapes. First, the models are used for shape discrimination, to constrain any segmentation framework by way of regularisation. Second, the SSMs are used for shape generation, allowing probabilistic segmentation frameworks to draw shapes from a prior distribution. The generative models also include novel methods to constrain shape generation according to information from both the image and user interactions.

The shape models are first evaluated in terms of discrimination capability, and shown to out-perform other shape descriptors. Experiments also show that the shape models can benefit a standard type of segmentation algorithm by providing shape regularisers. We finally show how to exploit the shape models in supervised segmentation frameworks, and evaluate their benefits in user trials.

Acknowledgements

I am very grateful to my supervisor Danny Alexander, for his encouragement and unwavering readiness to help throughout my PhD. I also thank my second supervisor Simon Prince, for the guidance he gave and an insight in particular into Gaussian Processes. I also appreciate discussions relevant to chapters 7 and 8, enjoyed with Lewis Griffin and Cedric Archambeau.

For invaluable help with neuroimaging datasets and kind efforts in performing software trials I wish to thank colleagues at the Institute of Neurology (IoN) Dan Tozer, Tom Hayton, Leonora Fisniku, Julian Furby and Joe Swanton. I also thank Jo Barnes and Jo Foster at the dementia group of IoN and Alastair Reid at the MRC Clinical Sciences Centre, Hammersmith Hospital, for demonstrating modes of interaction in the software packages 'DispImage', 'MIDAS' and 'Analyze'.

For their various roles as volunteers, both in performing experiments and reflecting on the teachings of Jeremy Bentham, I wish to thank colleagues Baptiste Allain, Yu Bai, Matt Hall, Min Kim, Martin Lillholm, Andreas Mang, Wallizada Moheb, Gemma Morgan, Laura Panagiotaki, Martin Parsley, Chris Senanayake, Kiran Seunarine, Bernard Siow and Xiahai Zhuang.

This project was funded by the EPSRC.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Image segmentation identifies regions and/or boundaries in an image to turn pixel intensities into semantic information. This aids interpretation of an image and is necessary as a pre-processing step in many image analysis techniques. Segmentation is a fundamental challenge to image processing, encompassing numerous approaches, goals and requirements depending on the application. For many applications, segmentation is challenged by poor image quality and variable shape of the region of interest (ROI). This is true of many medical imaging tasks, where images are typically monospectral, low contrast and noisy. Along with these difficulties, medical tasks such as disease diagnosis and monitoring, treatment planning and image guided surgery have high demands in terms of accuracy, precision and generalisation.

Automatic segmentation often fails to generalise beyond specific applications and strict experimental conditions. As a result, and due to the user's desire for control, manual segmentation is commonplace. However, fully manual methods are labour intensive and the results are prone to variability. Our aim is to balance automation with user control by finding new ways to learn from the texture and shape of pre-segmented regions, along with the efficient use of on-line supervision in a segmentation algorithm.

1.1 Background

Segmentation has received various definitions, influenced by its many applications. A standard image processing text states that segmentation 'subdivides an image into its constituent regions or objects' [1]. Definitions in the computer vision literature include 'partitioning of a given image into a number of homogeneous regions according to a given critical' [2], while the medical image analysis literature includes definitions such as 'identification and quantitation of tissues and organs' [3]. Similarly, there are numerous approaches to segmentation, depending on the application and the form of results sought.

1.1.1 Pros and cons of fully automatic and manual methods

To understand the motivation for semiautomatic methods, it is useful to look at the two extremes of fully automatic and fully manual methods. The project can then strive to maintain the benefits, and suppress the disadvantages, of each.

Fully automatic methods can produce the same result for repeated segmentations. The removal of variability can make automatic methods more reliable for use in longitudinal studies, but places more demand on the results themselves. In theory, the results of automatic methods are not affected by the

user. However, in practice the unique segmentation presented by automatic procedures often requires post editing before the user is satisfied with the results. Automatic methods often compensate for the lack of user information with a relatively high amount of pre-processing such as classification of image texture or multispectral data. Pre-processing often imposes demands upon the data, such as multispectral acquisitions, calibrated dynamic ranges, isotropy or alighment in a standardised coordinate system. The increased pre-processing can also lead to the loss or smoothing of information and impractical computation times.

The fully manual method of freehand boundary delineation requires maximal user input, leading to three main disadvantages. First, the inevitable subjectivity of a single user leads to inter-operator variability. Second, human imperfection and the 'user fatigue' arising for longer tasks leads to intra-operator variability when the same user repeats a segmentation task on the same region. Third, the manual approach introduces demands on an operator, as they must have expertise in the segmentation task and be able to spare the time to perform manual segmentation.

1.1.2 Deformable contour models

Methods of segmentation can be divided into thresholding, region-based, boundary-based and hybrid [4]. This project focuses on boundary-based methods, which model a region outline as a 2-dimensional parametric or geometric contour. This general family of methods will be referred to as *deformable contour models* (DCMs) and, while their approaches vary widely in the literature, their frameworks share a common set of components listed in Table 1.1. These components are separated to aid discussions throughout this thesis, but we note that the list is not absolute and components are naturally linked. For example a shape model ($\mathcal{C}_3$) might naturally yield a measure of agreement between a contour model and shape class, which can be used in an objective function ($\mathcal{C}_4$). Certain sections of this thesis refer explicitly to components in table 1.1 (with labels $\mathcal{C}_\#$) to help place certain topics in the context of a segmentation framework.

1.1.3 Boundary ambiguity and variable shape

This project addresses segmentation tasks confounded by boundary ambiguity and variable shape. These tasks pose particular challenges to a segmentation framework as the image model ($\mathcal{C}_2$) should identify boundaries and a shape model ($\mathcal{C}_3$) generally requires some predictable features of the region's shape.

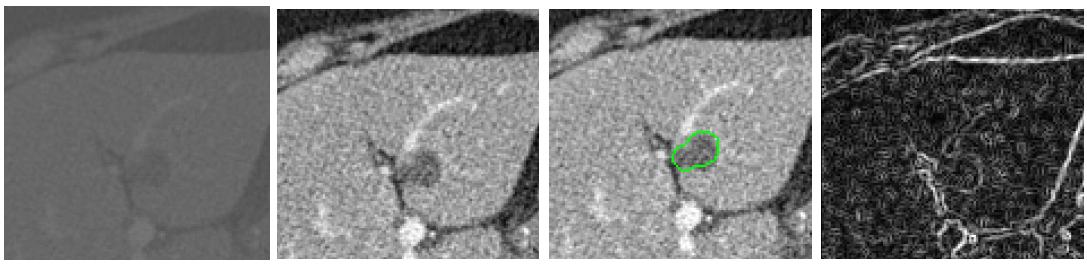
Boundary ambiguity occurs due to weak intensity gradients, the presence of texture and 'clutter', and similarity of intensity histograms between ROI and surrounding data. These factors are common in medical imaging applications because of physical limitations of the imaging technology and the fact that a region, which is distinguished from its surroundings by its pathology, function, or other semantic attribute, may not be distinct in its signal response. An example of boundary ambiguity is the outline of a liver tumour in a CT slice such as that in figure 1.1. The tumour in (a) is hardly visible and the edge map in (c) shows that tumour edges have low gradient magnitude and are barely discernible from nearby clutter.

Variable shape arises, for example, when shapes belonging to the same class do not share global shape properties such as spatial correspondence between boundary points. Figure 1.2 illustrates variable

Label	Component	Description	Examples in the literature
$\mathcal{C}_1$	Contour representation	Choice of parametrisation and coordinate system. Discrete or continuous representation of 1-D object.	Ordered list of N pixel coordinates along a discretised boundary [5] or zero-level set of a continuous function [6].
$\mathcal{C}_2$	Image model	Interpretation of pixel information. Often uses supervised classification.	Boundary measures derived from intensity gradient, eg 'Gradient Vector Flow' (GVF) [7]. Region-based intensity or texture classification [8].
$\mathcal{C}_3$	Shape model	Parametrised physical or statistical model of high- or low-level shape information. Also used for shape classification and object recognition.	Local smoothness, 'tension' and 'stiffness' [5]. Principal Component Analysis based on 'Point Distribution Model' (PDM) [9]. Geometrical descriptors [10].
$\mathcal{C}_4$	Objective function	Probability or 'energy' associated with a contour, based on its shape and position in the image model. Often combines $\mathcal{C}_2$ and $\mathcal{C}_3$.	Weighted sum of internal 'energy' and proximity to high gradient magnitudes in the image [5].
$\mathcal{C}_5$	Deformation mechanism	Local or global perturbations of a candidate contour. May use or conform to shape a model.	Stochastic sampling of feasible boundary sections [11], moving contour points throughout a search window [12].
$\mathcal{C}_6$	Optimisation scheme	Fitting a contour to a region boundary by maximising a probability or minimising an energy functional ($\mathcal{C}_4$).	Variational methods [5], greedy algorithm [12] or Monte Carlo methods [2].
$\mathcal{C}_7$	Modes of interaction	Information provided by mouse cursor in initialisation, supervision and post editing.	Locating 'volcano' points for a contour model to avoid [5]. Marking example data to include inside a region [13].

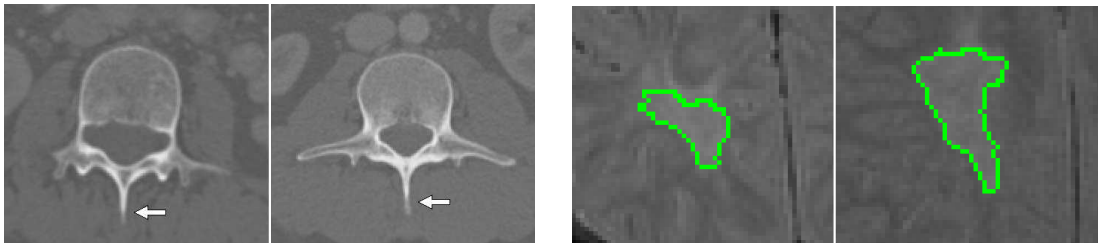
Table 1.1: Components of a deformable contour segmentation framework.

shape along with a counter example of regions with correspondence points. In (a) a class of shapes, healthy vertebra, exhibits correspondence between the recurring boundary feature (*spinous process*) present in all examples of this class. (b) shows a different class of shapes, namely multiple sclerosis lesions, which do not share obvious global properties. Many medical regions of interest have such



(a) (b) (c) (d)

Figure 1.1: (a) Cropped slice from an axial CT image showing a tumour region in the liver. (b) The same image after contrast enhancement. (c) The boundary of the tumour (green), delineated manually by an expert. (d) Gradient magnitude of (a), shown in grey scale from black (zero) to white (maximum).



(a) (b)

Figure 1.2: (a) Cropped slices from two separate axial CT images, each showing a cross-section of a human vertebra. Arrows indicate the *spinous process*, a recurring boundary feature. (b) Cropped slices from mid-axial regions in MRI scans of two separate human brains. Outlines (green) show a multiple sclerosis lesion in each case, as delineated by an expert rater.

variable shape due to the complex boilogical processes that form them.

1.1.4 Medical and biological imaging applications

Segmentation is necessary as a pre-processing step in medical image analysis techniques such as surgery planning, diagnosis, image registration and disease monitoring to mention but a few. This work uses data from multiple sclerosis lesion and liver tumour regions. These ROIs are often segmented manually or semiautomatically, as combinations of boundary ambiguity and shape variability make automation difficult.

Multiple sclerosis is a chronic disease that causes the destruction of myelin and loss of axons throughout the central nervous system. Demyelination leads to functional and sensory impairment but is reversible. Axonal loss, while less common, can cause permanent neurological disorder. Magnetic Resonance (MR) is the predominant imaging tool used for *in vivo* studies of MS. T2- and PD-weighted MR images are commonly used to detect and measure the white matter lesions (WML) that characterise the disease. Precise segmentation or delineation of lesion boundaries leads to measures of lesion load (total volume) and morphological information. Segmentation therefore aids studies of the disease itself, and enables more informative monitoring of changes over time, for example in response to drugs in

clinical trials. The main challenge to the future of conventional MR in MS studies is to provide better discrimination of lesions in images [14].

MS lesions have a wide variety of shapes and sizes. The morphology and internal structure of lesions vary over time due to the complex histology of the disease [15]. A single brain can also contain multiple lesions at different stages of their evolution [16]. For a given stage in the disease, the pathology of a lesion also varies between patients [17]. MR images exhibit spatial intensity non-uniformity, due to magnetic field inhomogeneities within the field of view, leading to apparent differences between lesions in different areas of the brain. Scanner inconsistencies can also cause the same lesion in the same patient to be presented differently in successive scans using the same machine [18].

In practice, MS lesion contouring routinely calls for manual input [19, 20, 21]. At London's Institute of Neurology (IoN), four people who routinely segment lesions reported that 40% to 60% of automated contouring using the tool in [22] required subsequent manual editing, of which roughly half required complete replacement with a freehand contour [23, 24, 25, 26].

Liver tumours are among the most common tumours affecting adults, as the liver is the largest internal organ of the human body and its risk from abnormal cell growth is increased by lifestyle factors such as alcohol consumption. Liver tumours also occur metastatically in relation to other common diseases such as colorectal cancer [27]. Metastatic liver cancer has a median survival rate of less than one year if left untreated [28]. Surgical resection is the most common form of treatment for metastatic liver tumours. This places a high demand on improved practices in image guided surgery, as reduced resection margins leads to more patients eligible for the treatment and higher success rates. Radiotherapy is also used to treat liver cancer (malignant tumours), which in turn demands accurate localisation of the radiation dose. Surgery planning, guidance and radiotherapy all benefit from accurate tumour segmentation in medical images.

Livers are commonly studied using abdominal X-ray CT imaging. However, tumours do not appear with distinct intensity or texture characteristics and their boundaries are difficult to see even by the human eye. Liver tumours pose one of the biggest challenges to the fields of tissue classification and boundary extraction in medical image analysis [29].

To emphasise the wider applicability of the ideas presented in this thesis, we briefly mention here some other areas that our investigations do not consider, but where manual segmentation in 2-dimensions is commonplace. First, biomedical research is also seeing an increase in live cell imaging thanks to advances in phase-contrast microscopy [30]. This imaging modality produces 2-dimensional data, wherein the cell boundaries are discontinuous, have badly localised edges spanning several pixels, and are not known to benefit from existing shape models [31].

In another area, novel neuroimaging techniques can distinguish individual synaptic structures such as axons using data from state-of-the-art scanning electron microscopy [32]. The technology can image tissue volumes with voxel dimensions of tens of nanometers, wherein the segmentation of axons would allow the reconstruction of complete neuronal structures at a cellular level. For this application, boundary ambiguity results from the presence of 'clutter' and the proximity of neighbouring axons. State-of-the-

art segmentation uses region-based 'graph-cuts' [33] or geometric active contours [34]. However, in both approaches, the boundary ambiguity problem leads to errors that are rectified by post editing, which is fully manual in the case of [34]. Also, neither this nor the graph-cut approach of [33] use prior shape models and in the latter case, the algorithm does not readily incorporate shape priors if such knowledge were available

Alzheimer's disease, the most common type of dementia, is another application area calling for improved supervised segmentation. Hippocampal segmentation in MRI neuroimaging plays an important role in the diagnosis, monitoring and studying of the disease. Hippocampi can be seen to shrink over time, serving as indicators of brain atrophy, or loss of grey matter volume, associated with the disease [35]. At London's Institute of Neurology (IoN), hippocampal segmentation uses the software in [36]. Depending on the extent of atrophy, a single hippocampal segmentation can take an experienced user up to 45 minutes [37].

1.2 Problem Statement

This project focuses on applications where the region of interest is poorly defined due to variable shape and poor image quality. This applies to a family of medical ROIs referred to in [38], as

> " natural objects, such as those found in biomedical images, whose diversity and irregularity of shape make them poorly represented in terms of fixed features or form".

For these applications we maintain that

> *Lemma 1*: There is no single perfect *result* of segmentation [39], and
>
> *Lemma 2*: The perfect *method* of segmentation allows for different outcomes, where a human expert ultimately dictates the result.

Adopting *Lemma 2* rules out fully automatic segmentation for our purposes. We aim instead to develop semiautomatic tools that increase user control, but work *with* the user to reduce the demand on them. We adopt the requirements stated in [40, 41, 42], of

> *Requirement 1*: providing as complete control as possible to the user, and
>
> *Requirement 2*: minimising user involvement and total user time necessary without compromising precision and accuracy.

The problem falls inside the broader area of region segmentation in medical imaging, where improvements for accuracy, user demand and repeatability are likely to be made for years to come. Manual or supervised segmentation is likely to survive even as automatic methods improve in the future. As well as the need for user input implied by *Lemma 1* and *Requirement 2*, research into fully automatic segmentation depends in turn on manual/supervised practices in two ways. First, methods that use machine learning, such as statistical shape models or supervised texture classification, require training data defined by expert observers. Second, methods of performance evaluation often assume some form of 'ground truth', which is ultimately created or approved by human experts.

1.3 Approaches

Requirements *1* and *2* are conflicting and need to be balanced. We believe that the requirements are balanced by efficient interaction and prior knowledge. To this end we develop dynamic contour models with on-line supervision by novel modes of interaction along with off-line machine learning to maximise the prior knowledge available. To address boundary ambiguity we develop a support vector machine (SVM) classifier for use as a generalised image observation model. To address variable shape and further assist the weak boundary problem we develop novel statistical shape models based on time series analysis. We build the texture models into a boundary tracking framework and the shape models into various novel closed contour DCMs. These segmentation frameworks introduce novel modes of interaction for efficient run-time supervision.

1.4 Constraints

The focus on user interaction leads to three key constraints on this work. First, the study is mainly concerned with 2-dimensional segmentation, but suggests 3-dimensional extensions. We justify this constraint as follows:

- The aim is to semiautomate those scenarios mentioned in section 1.1.4, where manual delineation is common practice. Semiautomatic methods 'do some of the work' on behalf of the user, by pre-empting or mimicking their actions. By viewing manual delineation as a starting point to be improved by semiautomation, we constrain the method to the same, 2-dimensional domain.

- During a segmentation procedure, it is only practical to visualise a cross-section containing the ROI at any one time. Although 3-dimensional surface rendering is possible in contemporary image analysis software, this is used to display the *results* of segmentation.

- Common medical imaging modalities such as MRI and CT produce images that are isotropic in 2-dimensions, having different (lower) resolution in the third dimension. It is therefore often sensible to segment the whole of a 3-dimensional region by the successive segmentation of cross-sections in the 2-dimensional plane of higher resolution.

- A 2-dimensional framework allows more natural interactivity, as a user is most likely accustomed to drawing 2-dimensional objects, either on computer displays or with pen and paper.

- For some applications such as MS lesion contouring, the physically thin ROIs might only appear in one image plane or otherwise show little correspondence between slices.

The second constraint concerns multiple regions. Where more than one ROI exists in a given image, a user only pays attention to one at a time. This excludes any need for simultaneous segmentation of multiple regions. The third constraint concerns region *detection*. We assume that user-initialisation removes the need for automatic detection of a region of interest.

1.5 Contributions

This project advances the field of supervised segmentation by making the following key contributions.

SVM texture classification

We develop binary SVMs for generalised texture classification in applications suffering boundary ambiguity. Experiments motivate the use of SVMs to provide observation models ($\mathcal{C}_2$).

Tracking ambiguous boundaries

We develop and test interactive boundary tracking methods for segmentation by boundary tracking. The contributions are extensions to the 'jetstream' algorithm in [43], to use texture classification in the observation model ($\mathcal{C}_2$), adapted deformation mechanisms ($\mathcal{C}_5$) and novel modes of interaction ($\mathcal{C}_7$).

Time series shape models

We introduce nonlinear time series methods to the field of global shape modelling ($\mathcal{C}_3$). We demonstrate using Langevin and Gaussian Process methods for modelling 2-dimensional regions. Experiments motivate the use of nonlinear time series analysis for shape modelling and their extension for use in semiautomatic segmentation.

Time series segmentation frameworks

We introduce the use of novel time series models as shape priors in interactive segmentation by deformable contour models. We use radial time series contour representations ($\mathcal{C}_1$) and design frameworks that exploit various techniques of Langevin and Gaussian Process models. Discriminative use of time series models leads to shape regularisation by incorporating global shape information into probabilistic objective functions. Generative use of time series models leads to new DCM frameworks that build the shape information into the deformation mechanism ($\mathcal{C}_5$) and optimisation scheme ($\mathcal{C}_6$). We show how to exploit user interactions in setting key model parameters upon initialisation as well as, in the case of Gaussian Process models, conditioning the model during runtime. Experiments demonstrate the success of the models in these various roles.

1.6 Overview

The remainder of this thesis is organised as follows. Chapters 2 to 5 form a review of relevant literature, divided into methods with the general goals of segmentation and shape modelling (chapters 2 to 3) and other background material used in the project (chapters 4 to 5).

The review of segmentation and shape modelling methods includes both 'state-of-the-art' and other approaches that bare relevance to this work. Chapter 2 focuses on interactive boundary-based segmentation methods and highlights their application to tasks involving boundary ambiguity and variable shape. Chapter 3 focuses on global shape models with a machine learning element, highlighting when these are used in segmentation frameworks and their applicability to medical tasks or ROIs with variable shape.

The review of background material covers ideas that this work draws from in its methodological

approaches, including ideas that are not directly related to image segmentation. Chapter 4 describes supervised classification with emphasis on kernel methods, which are treated as a machine learning approach to the image model ($\mathcal{C}_3$) for use in segmentation. This topic is relevant to our methods of discriminating regions and their boundaries in texture images. Chapter 5 reviews chosen topics in the field of nonlinear time-series analysis, which are treated as a machine learning approach to shape modelling ($\mathcal{C}_3$) for use in segmentation.

Chapters 6 to 8 report on the research undertaken. We use machine learning techniques to increase prior knowledge from both image and shape to reduce the amount of on-line supervision necessary (*Requirement 2*). We build these into segmentation frameworks along with novel modes of interaction that work with the underlying segmentation algorithm or shape model to increase user control (*Requirement 1*). Chapter 6 investigates a machine learning classifier to identify boundaries for use as an image model ($\mathcal{C}_2$) and incorporates modes of interaction ($\mathcal{C}_7$) added to the jetstream algorithm in [43]. Chapter 7 introduces nonlinear time series methods for statistical shape modelling. Chapter 8 introduces novel segmentation frameworks based on the shape models, using both discriminative and generative methods.

All demonstrations and experiments use both synthetic and medical data. For texture classification we use synthetic textures derived from images in the 'Vision Texture' database [44] along with textures present in MR images. For ROI segmentation we use both medical images and synthetic images, where synthetic images allow more robust performance evaluation due to the availability of 'ground truth'. Synthetic texture images in chapter 6 recreate the boundary ambiguity problem by using textures that share first- and second-order histogram statistics with medical ROIs and surrounding tissue. Synthetic contrast images in chapter 8 recreate shape variability by using real tumour and lesion contours as region boundaries. .

Chapter 2

Deformable Contours for Interactive Segmentation

This chapter reviews current segmentation methods, paying particular attention to methods that are applicable to the problem stated in section 1.2. According to *Lemma 2* a user must be able to influence segmentation, so this review emphasises interactive techniques. The need to balance requirements *1* and *2*, of maximal user control with minimal user demand, leads to an emphasis on where methods make efficient use of interactions and incorporate machine learning. In addition, the review is weighted toward methods belonging to the family of *deformable contour models* (DCMs), as outlined in section 1.1.2, which are suited to the project for two main reasons. First, these models readily allow the user to visualise and interact with the various contour representations directly. Second, the contour representations facilitate internal and global shape constraints, helping to overcome challenges of boundary ambiguity and variable shape. We further weight the review toward 2-dimensional practices as justified in section 1.4. However, we naturally consider a method's extension to 3 dimensions as an asset, and discuss these extensions in section 2.5. Where a method uses a global shape model ($\mathcal{C}_3$), this is mentioned in passing and revisited in a more detailed review of shape models in chapter 3.

The contour representation ($\mathcal{C}_1$) of boundary-based methods can be geometric or parametric. Geometric contours are smooth and implicit, while parametric contours comprise explicit points, which are joined either in a spline representation, giving smooth curves (eg. [45, 46]), or by linear interpolation. The project emphasises parametric contours as they address our goals in two ways. First, making discrete boundary elements visible to a user facilitates efficient modes of user interaction. Second, parametric contours allow certain boundary tracking and shape modelling methods, which we extend in chapters 6 to 8.

This project does not use region-based methods for the reasons given above, in particular it is harder to incorporate global shape models into these methods. However, the state of the art of interactive segmentation includes region-based methods such as those based on graph-cuts, Markov random fields and region growing algorithms, and a review of interactive segmentation should not overlook these. We review these methods in terms of their modes of interaction, where this gives insight into more general issues of supervised segmentation.

The remainder of this chapter is organised as follows. Section 2.1 looks at closed contour models, divided into geometric contours 2.1.1 and parametric contours 2.1.2. Section 2.2 reviews a certain type of parametric contour, which start as an open contour and track a closed region boundary. Section 2.3 goes into more detail on where these and other segmentation methods make use of run-time interactions. Section 2.4 elaborates on how deformable contour frameworks have been evaluated and section 2.5 discusses the main findings in the review, summarising the implications for the present research goals.

2.1 Closed Contour Models

This section reviews segmentation frameworks that use closed contour models. We first discuss a geometric contour approach in section 2.1.1, which introduces some common aspects and challenges of deformable contours, while the rest of the review is weighted toward parametric contours.

2.1.1 Level sets

Geometric Active Contours or 'level sets', were introduced by Malladi *et al* [6] and developed by Caselles *et al* [47]. Level set methods define a contour as a continuous function of the image field, discretised at pixel level. The task is to find the continuous function for which a constant value (the zero level set) coincides with a region boundary. Classical level sets evolve under the influence of edge information and internal energy minimisation. For an evolving function $\phi(\mathbf{x}, t)$ of the field $\mathbf{x} = (x, y)$, the fundamental equation of level sets is [48]

$$\frac{\partial \phi}{\partial t} = V(\mathbf{x}) \times (\mathbf{x} - \phi \nabla \phi), \tag{2.1}$$

where $V(\mathbf{x})$ is the velocity of the points on the zero-level curve, acting in the direction normal to the curve at (x, y) with magnitude governed by local curvature. At any time t there is a set of n solutions $(x_0, x_1, ...x_{n-1}), (y_0, y_1, ...y_{n-1})$ for which $\phi(\mathbf{x}) = 0$. These are the n coordinates of a discrete contour in the image frame, at the interface of a propagating surface and the image. Equation 2.1 is iteratively solved for ϕ, until the propagating 'front' finds the region boundary.

As with many deformable contours, the classical framework based the image observation model ($\mathcal{C}_2$) on intensity gradient. This causes problems in low contrast or noisy images wherein boundaries defined by gradient are ambiguous. For example, a contour may fail to recognise weak edges and extend into areas outside the ROI - an artefact known as 'leaking' (see eg. [49]). Conversely, stronger gradients at nearby disconnected features or 'clutter' may erroneously attract the contour.

Leaking is avoided by the use of 'regularisers' in the velocity term, such as region information or a global shape model. There have been numerous studies into the inclusion of shape priors in a level set framework [50, 51, 52, 53]. In a different approach to spatial regularisation, Shen *et al* [54] present a method specific to segmenting individual vertebrae in images of the spine. The authors identify a centre-line along the spinal canal and define planes that intersect the centre-line and lie between vertebrae. These planes constrain the level sets so that individual closed contours do not merge or overlap.

More recent improvements to level sets involve developing new, stochastic optimisation schemes ($\mathcal{C}_6$). Two examples use stochastic calculus [55] and Bayesian probability [56]. The 'stochastic active

contours' in [55] minimise an objective function by solving a stochastic partial differential equation. The results demonstrate, without proof, that this method does not converge to local minima. However, the contour model fails to delineate concave boundary sections in some of the results shown. The contour also appears jagged over long, straight sections where a region boundary itself is smooth. This latter artefact may affect medical applications if segmenting long, thin anatomical structures such as vessels or aorta.

The probabilistic level sets in [56] use Markov Chain Monte Carlo (MCMC) optimisation. The MCMC algorithm is a general optimisation method, and as such will be seen in other parts of this thesis. In [56] the algorithm seeks to draw sample curves C from a distribution $p(C|I)$, given the image data I, where this distribution is intractable. Consider the zero level sets C and Γ as continuous curves parametrised by arc length $s \in [0, 1]$. Proposal curves are generated for the next time step $(t + 1)$ by the prediction

$$\Gamma^{(t+1)}(s) = C^{(t)}(s) + r^{(t+1)}(s)N_{C^{(t)}}(s) \tag{2.2}$$

where $r^{(t+1)}$ is a random perturbation field at time $t + 1$ and $N_{C^{(t)}}(s)$ is the normal to the curve C at time t. The perturbation is constructed from Gaussian noise convolved with a smoothing kernel, and also incorporates a curve length penalty to ensure that a curve initialised at the image boundary will shrink in the absence of external forces. Each proposal has an associated acceptance probability $a(\Gamma^{t+1}|C^t)$ that uses probabilities conditional on the previous state $q(\Gamma|C)$ and the image priors $p(\Gamma|I)$. The authors calculate the acceptance probability by the Hastings ratio

$$a(\Gamma^{t+1}|C^t) = \frac{p(\Gamma^{(t+1)}|I)q(C^t|\Gamma^{(t+1)})}{p(C^{(t)}|I)q(\Gamma^{(t+1)}|C^t)}, \tag{2.3}$$

where $q(C^t|\Gamma^{(t+1)})$ is the *reverse* proposal distribution that must be approximated. After a reasonable number of iterations referred to as 'burn-in', the samples are equivalent to a set drawn from the unknown distribution $p(C|I)$.

The data driven probabilities $p(C, \Gamma|I)$ must incorporate an image model to favour level sets lying on edges or separating distinct segments. In a demonstration applied to medical ROI segmentation, the authors use supervised classification to segment prostate regions in MR images. These results demonstrate qualitatively the success of the method.

A major benefit of this kind of probabilistic framework, is that they give a distribution over likely results, in line with *Lemma 1*. Probabilistic observation models can tighten this distribution using external information provided interactively. In an elegant example of this, the tool in [56] allows the user to mark accepted sections of a contour model at intermediate stages. Evolution then continues with a conditional probability distribution, having zero variance on accepted sections of the curve.

2.1.2 Active contour models

Active contour models (ACMs) or 'snakes' introduced by Kass *et al* [5] are a family of parametric contour models. The contour representation is a set of points $\mathbf{s} = \{s_0, \ldots, s_i, \ldots, s_{n-1}\}$ along the arc-length s. The evolving model, at time t, is defined by $\mathbf{u}(s, t) = \{\mathbf{x}(s, t), \mathbf{y}(s, t)\}$. ACMs are characterised by an energy functional of the contour $E(\mathbf{u})$, which changes with the shape of the contour

and its position in the image. The energy is the objective function ($\mathcal{C}_4$), designed to be minimum when the contour outlines the ROI, and generally involves two terms

$$E = E_{\text{int}}(\mathbf{u}) + E_{\text{ext}}(\mathbf{u}), \tag{2.4}$$

where the 'internal' energy E_{int} constrains the shape of the contour and the 'external', energy E_{ext} drives the contour to align with region boundaries. Internal energy is traditionally calculated by two terms that measure local 'tension' and 'stiffness' of a contour respectively, given by

$$E_{\text{int}}(\mathbf{u}, t) = \int \alpha |\frac{\partial \mathbf{u}}{\partial s}(s,t)|^2 + \beta |\frac{\partial^2 \mathbf{u}}{\partial^2 s}(s,t)|^2 \, ds, \tag{2.5}$$

where the first increases with distance between successive contour points and the second penalises high curvature. Constants α and β allow relative weighting of tension and stiffness constraints. As ACMs use a discretised contour, the continuous integrals in equation 2.5 are approximated by finite element equations, summed over n boundary points.

External energy E_{ext} is the image observation model ($\mathcal{C}_2$) and, as with level sets, the classical framework uses the magnitude of the image gradient. The total snake energy becomes a weighted sum of three terms

$$E(\mathbf{u}, t) \approx \sum_{i=0}^{n-1} \left(\alpha |\frac{\partial \mathbf{u}}{\partial s}(s_i,t)|^2 + \beta |\frac{\partial^2 \mathbf{u}}{\partial^2 s}(s_i,t)|^2 - \gamma E_{\text{ext}}(s_i,t) \right), \tag{2.6}$$

where constants α, β and γ weight the relative influence of each term. In a snake framework the optimisation scheme ($\mathcal{C}_4$) must minimise the functional in equation 2.6. Kass *et al.* use variational calculus, which has several drawbacks summarised in [57]. Dynamic Programming [57] and the greedy algorithm presented in [12] offer common alternatives. The greedy algorithm involves iteratively searching local to each boundary point in turn, for a new position that reduces the total energy. An ACM evolving with the greedy algorithm can become stuck in an oscillatory state and never converge. This happens when information regarding the image and local shape properties are conflicting.

Snakes are sensitive to the weights α, β and γ in equation 2.6. Optimal pairs α and β depend on the shape and smoothness of the true boundary while the choice of γ depends on the reliability of the image observation model. These factors vary between images, reducing the generality of the method. The parameters can be tuned to the application at hand by user interaction as in the implementation of Jacob *et al.* [45]. However, fixed values might not be suitable for some medical applications where a ROI boundary has spatially varying smoothness or corners [58]. Some improvements to classical snakes allow adaptive energy weights [12] and [59].

The authors in [12] and [59] modify different components of the ACM framework. The method in [12] changes the objective function ($\mathcal{C}_4$) enabling the stiffness to vary around the contour. The authors set $\beta = 0$ at 'corners' identified by limits of local curvature and edge strength. The method in [59] modifies the framework by changing the optimisation scheme ($\mathcal{C}_6$). The authors use a local *minmax* search that simultaneously solves the minimisation of the contour energy with an optimisation of the energy term weights. The weights are first combined in a single parameter λ, which is allowed to vary with position along the contour as $\lambda = \lambda_0, \lambda_1..\lambda_{n-1}$. All λ_i are then varied independently, along with

the neighbourhood search of each s_i, and the method finds the unique *minimum* with respect to $\mathbf{u}$, of the *maximum* energy over all λ.

The tension and stiffness definitions above are not necessarily optimal for a given application. Perrin and Smith redefine tension based on the mid-point between neighbouring boundary points and curvature based on third derivative $\frac{\partial^3}{\partial s^3}$ in the 5-point neighbourhood centred on s_i [60]. This curvature definition successfully segments ventricles in MR brain images in [61].

As in the case of level sets, the classical snake framework was driven by gradient magnitude and subsequent improvements replaced this component of the framework ($\mathcal{C}_2$). A popular alternative to gradient magnitude is the 'Gradient Vector Flow' (GVF) introduced by Xu and Prince [7]. Computed by minimising a functional of the intensity field, the GVF gives a map of gradient vectors that can attract a contour over larger distances. The main benefit of this energy is that contours are less likely to traverse concave boundary sections. Another benefit is that the GVF reduces the need to initialise a contour close to the true ROI, hence demand on the user. The occurrence of concave boundary sections makes GVF a popular choice of observation model in medical image segmentation (eg. [62]). However, problems associated with false edges or clutter common in medical images are not addressed by the GVF.

Again, in common with level set frameworks, another family of observation models overcome leaking artefacts by incorporating region information. Region models were first introduced to ACM frameworks by Ivins and Porrill [63] and independently by Ronfard [64]. The 'Active Region Model' (ARM) of Ivins and Porrill is equivalent to an ACM in all but the image energy term in equation 2.6, which is replaced by a region model. Some authors use region models based on histograms of region and background intensity [8, 65, 66]. For multidimensional data the distributions can be used to derive the Mahalanobis distance from Gaussian joint probabilities. The Mahalanobis distance serves as a 'goodness' measure, of the agreement with training data, as demonstrated in [65] and more recently in [66], and assessed in detail in [67]. Ivins *et. al* [8] use the ARM to segment medical images, where the multidimensional data are made up of co-registered NMR and CT images. However, with the exception of hybrid imaging modalities such as PET/CT or multi-weighted MRI, this approach relies on numerical registration methods, which reduce the accuracy and resolution of boundary definition. When only monospectral data is available, some authors improve region models by calculating texture features derived, for example, from Laws masks [68] or grey-level co-occurrence matrices [69, 70]. In addition to an image model ($\mathcal{C}_2$), external energy can incorporate information provided by user interaction ($\mathcal{C}_7$). In their seminal work, Kass *et al* [5] suggest an extra term in equation 2.6, being a function of the distance to points located by the mouse cursor. This so called 'volcano' concept allows the user to guide the contour toward the target region by specifying points of repulsion.

2.1.3 Brownian strings

In 1997 Grzeszczuk and Levin introduced 'Brownian Strings' for interactive boundary extraction [11]. We discuss Brownian Strings as they use novel or uncommon examples of several components ($\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{C}_5$, $\mathcal{C}_6$ and $\mathcal{C}_7$). The work also highlights the way in which certain components are inextricably linked, and the difference between validation and performance evaluation.

As a contour representation ($\mathcal{C}_1$) the authors use connected 'cracks' between pixels inside and outside the region. This representation removes the ambiguity in the resulting segmentation, where 'boundary pixels' in the representations above may strictly belong to either region or background. In practice, contours are stored as a 2-dimensional array of cracks called a 'crack diagram', rather than a 1-dimensional list. This data structure retains the relative positions of each crack and allows constraints to avoid self-intersection of the contour.

The observation model ($\mathcal{C}_2$) exploits the contour representation. The authors derive a probabilistic boundary measure from training examples of crack diagrams aligned with true ROI boundaries. This boundary measure is based on the 2-dimensional histogram of pixel intensities on either side of a single crack. In the medical example of whole-brain extraction in MRI volumes, this supervised approach to an observation model makes use of interactive initialisation, as the histogram is built up from a boundary defined manually in a single slice and used to segment the brain in neighbouring slices.

The deformation mechanism ($\mathcal{C}_5$) is also specific to the contour representation. The authors present two stochastic methods to generate proposal boundary sections in the form of small crack diagrams. In one case, crack diagrams are simulated by novel geometric processes that ensure non-intersecting boundary sections. In the second case, valid crack diagrams are drawn at random from a library.

The contour evolves under a stochastic method of energy minimisation ($\mathcal{C}_6$), namely simulated annealing [71]. The authors state that this method is guaranteed to find the global energy minimum, although strictly speaking this is only true if the algorithm runs for infinite time.

As mentioned above, the algorithm is initialised by the user ($\mathcal{C}_7$) in one image slice for subsequent segmentation of another. Rather than manual drawing, the user must perform interactive, iterative thresholding. It is not clear whether this method is faster or more accurate than manual drawing. The initial contour has two roles. First, the user defined contour provides training data for the image model. Second, the authors use morphological operations of dilation and erosion to define an annular region, for use as a binary mask in neighbouring slices. The mask speeds up the simulated annealing by reducing the search space.

The results in [11] do not evaluate the performance of Brownian Strings. The authors merely demonstrate the method by segmenting synthetic and MR images. In the latter case the contour variously delineates the intended brain boundary and the boundary between grey and white matter in the cortex. The synthetic images also provide only qualitative results but are powerful validation tools. The authors create images designed specifically to pose the challenges of local minima, broken edges and clutter. Results clearly show the ability of the algorithm to overcome these challenges.

2.1.4 1D Cyclic Markov Random Fields

The 1D Cyclic Markov Random Field (1D-CMRF), first seen in [72], is one of a family of contour models that represent a contour in polar coordinates $\{\mathbf{r}, \theta\}$. The representation ($\mathcal{C}_1$) is a list $\mathbf{r} = \{r_0, \ldots r_t \ldots r_{N-1}\}$, of N radial distances from a fixed point inside the ROI to its boundary, separated by angular increments $\theta = \{\theta_0, \ldots \theta_t \ldots \theta_{N-1}\}$. This representation will be referred to hereafter as a *radial time series*. The 1D-CMRF model treats the radial time series as a vector of N discrete

random variables r_i, and each radius r_i as a site in a Markov Random Field (MRF). After choosing a point inside the region, the 1D-CMRF is defined by

$$\Pr(\mathbf{r} = \rho) \geq 0 \tag{2.7}$$

and

$$\Pr(r_i = \rho_i | r_j = \rho_j, j \neq i) = \Pr(r_i = \rho_i | r_j = \rho_j, j \in \mathcal{W}_i), \tag{2.8}$$

where $\rho = \{\rho_0, \ldots, \rho_i, \ldots, \rho_N\}$ is a possible configuration for $\mathbf{r}$, ρ_i is a 'hidden variable' or sample point for r_i and $\mathcal{W}_i$ is a 'clique' or neighbourhood of i ($\mathcal{W} = \{i-1, i+1\}$ for the example in [73]).

Equation 2.8 simply states the Markovian property that the probability for a given $r_i = \rho_i$ is conditional only on a (small) neighbourhood. MRF methods follow on from the Hammersley Clifford theorem [74] which states that if equation 2.7 holds then the joint probability $\Pr(\mathbf{r})$ is uniquely determined by conditional probabilities in equation 2.8 and that these follow a Gibbs distribution, i.e. $\Pr(\mathbf{r}) \propto \exp[-\sum_{j \in \mathcal{W}_i} U_j]$ where U_j is an energy function or 'clique potential' that embodies the *a priori* information, combining image and shape priors. In [73] the clique potential is a weighted sum of both 'low-level' and 'high-level' information. The high-level information is a crude statistical shape model, revisited in the next chapter. The low-level information is a weighted sum of a smoothness term and a 'step' term designed to align the radial time series with points of high image gradient. The smoothness term penalises local variation of radial distances inside a small angular window (the clique) and can be considered a re-formulation of the stiffness term of a classical ACM.

Subsequent work by [75] and [76] present 1D-CMRF models to extract contours of the left ventricle in X-ray angiography and ECG cardiology respectively. The different applications call for different clique potentials and likelihood functions ($\mathcal{C}_2$), and the authors also choose different optimisation schemes ($\mathcal{C}_6$). The clique potentials are all low-level according to the distinction above, imposing smoothness of different derivative-order by using neighbourhoods $\mathcal{W}_i$ of different sizes. The image observation models reflect the characteristics of the different image modalities but all assume an intensity difference between the region and background. Experiments in [76] use the segmentation method to derive the secondary results of ventricular volume and wall thickness and show that results agree with the same measures derived from manual segmentation. Qualitative results of ventricle segmentation in [75] further support the use of 1D-CMRF in medical images but do not allow for comparison.

In a novel adaptation of the 1D-CMRF, Martín-Fernández and Albertola-López define the radial time series as a list of radial *perturbations* about a 'mean contour' [77, 78]. Theirs is a supervised framework, wherein the mean contour is defined manually by the user. This is an example of making efficient use of interactivity, as the initial contour only approximates the ROI and is therefore fast to produce, but provides high-level information that is integral to the contour model. The authors demonstrated acceptable results for whole-organ segmentation of kidney regions in ultrasound images.

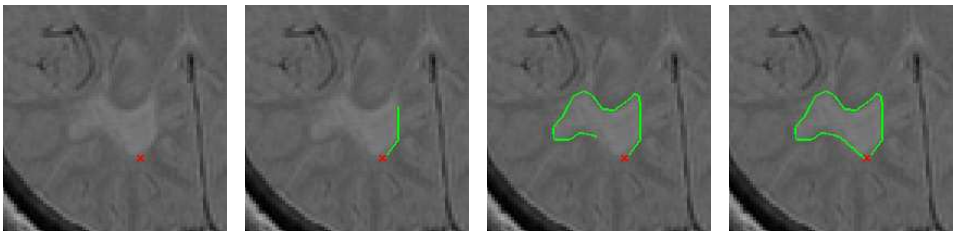
The polar representation leads to certain advantages and disadvantages of the 1D-CMRF. As pointed out in [72], the representation requires a 1-dimensional array of size N, saving considerable computation compared to region-based MRFs (see eg. [79, 80, 81]). The region-based counterpart operates on a

random field of size $W \times H$ where W and H are the width and height of the image. A minor drawback of the 1D-CMRF is that a point near the centre of the ROI must be identified to fix the polar coordinates in the image frame. In [72] the authors use an automatic detection procedure, which introduces extra computation time possible of errors, especially for low contrast or otherwise badly defined regions in medical images. However, as we will see in chapter 8, sufficient estimates of a region centre can be provided with little effort by user interaction.

The main drawback of the polar representation is the assumption that each radius intersects with the boundary only once. By definition, this limits the approach to model regions that are 'star-shaped' [82]. However, some applications naturally involve ROIs that are inherently star-shaped. In these cases other authors have chosen star-shaped representations of region boundaries. Examples in the biomedical field are tumours in PET images [83], kidney and pelvis regions in echography [77] and the left ventricle in cardiac images [75, 76].

2.2 Boundary Tracking Methods

This section describes deformable contour models that *track* a region boundary using open contours. Starting from a point on the boundary, an open contour progresses around the ROI and forms a closed contour if the whole boundary is tracked, as illustrated in figure 2.1.



(a) (b) (c) (d)

Figure 2.1: Schematic diagram of boundary tracking methods, illustrated on a region (MS lesion) in an axial MR image of the brain. (a) A single boundary point (red) is located. (b) Starting from the boundary point, a contour model progresses around the region, here in an anticlockwise direction. (c) As the algorithm continues the contour approaches the starting point. (d) In this case, the algorithm achieves a closed contour.

An appropriate starting point could be detected automatically or defined interactively by a point or *anchor* located by the the user. Boundary tracking methods are distinct from other open contour models by their deformation mechanism ($\mathcal{C}_5$). Boundary trackers deform at one end as the open section grows in arc-length.

2.2.1 Active testing and particle filtering

The contour representation ($\mathcal{C}_1$) in a boundary tracking framework is a chain of successive elements $\{\mathbf{x}_0, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_N\}$, where $\mathbf{x}$ is the position vector $\mathbf{x} = \{x, y\}$. One family of boundary tracking algorithms treats the contour as the result of N decisions, made to follow the boundary at each *step*. The

earliest example of this is the 'Active Testing' algorithm developed by Geman and Jedynak [84, 85]. The algorithm iteratively appends the contour with straight line sections of equal length. Each iteration must choose the angle, from a discrete set, between the current section and the next.

If the contour model is made up of N sections, and each decision chooses from three directions $\{-\varphi, 0, +\varphi\}$, the resulting 'decision-tree' involves 3^N potential outcomes. Ultimately, the segmentation task is to choose the outcome that is most likely to track the boundary. Active testing is a general technique for estimating the true 'hypothesis' $\mathbf{x}$ whilst only considering a fraction of the possible outcomes. This involves discarding sections of the tree at any given decision, thus reducing computational complexity. The ternary decision that selects from $\{-\varphi, 0, +\varphi\}$ involves testing candidate directions (or pixels in those directions) against a prior observation model ($\mathcal{C}_2$). The example in [85] uses a filter based on a statistical model of the relative intensity between pixels on and off the desired boundary. The outcome of a test is to accept one direction while the other two are rejected, which discards subsequent branches in the decision tree. Acceptance is based on a statistical test derived from information theory. The authors demonstrate using both 'entropy testing' rule and a maximum likelihood estimator. The authors demonstrated active testing for the application of tracking roads in satellite imagery. However, as noted in [85], the filters designed to distinguish roads from background can be replaced to generalise the algorithm for other applications.

Active testing can alternatively be performed using Monte Carlo rejection sampling in place of the statistical tests above. The resulting 'particle filter' algorithm was first used for multi-dimensional object tracking in Isard and Blake's 'CONDENSATION' algorithm [86]. In one dimension, the same algorithm extends to the task of tracking boundaries in images. The resulting algorithm, introduced as 'jetstreams' by Perez *et al.* [43] stores a set of M contours or 'particles' $(\mathbf{x}^m_{0,\ldots,i})_{m=0,\ldots,M-1}$. Tracking is based on the iterative computation of posterior densities over the next step at $\mathbf{x}_{i+1}$. Given image data $\mathcal{D}(x, y)$ the posterior probability for the next step is given by

$$p_{i+1}(\mathbf{x}_{0,\ldots,i+1}|\mathcal{D}) \propto p_i(\mathbf{x}_{0,\ldots,i}|\mathcal{D}) \times q(\mathbf{x}_{i+1}|\mathbf{x}_{i-1,\ldots,i}) \times l(\mathcal{D}(\mathbf{x}_{i+1})), \tag{2.9}$$

where $q(\mathbf{x}_{i+1}|\mathbf{x}_{i-1,\ldots,i})$ is the prior distribution over steps and $l(\mathcal{D}(\mathbf{x}_{i+1}))$ is the likelihood that a contour section lies on the region boundary, given the data. This distribution is defined over the continuous angle φ, made between $\mathbf{x}_{i-1,\ldots,i}$ and $\mathbf{x}_{i+1}$, making $\mathbf{x}_{0,\ldots,i}$ a Markov chain.

The jetstream algorithm computes p_{i+1} in three stages. A *prediction* stage randomly selects M locations for the next point $\mathbf{x}_{i+1}$, from the normal prior angular distribution $q(\mathbf{x}_{i+1}|\mathbf{x}_{i-1,\ldots,i}) = q(\varphi) = \mathcal{N}(0, \sigma_\varphi)$. By choosing angles distributed about zero, the jetstream maintains a smoothness constraint governed by the parameter σ_φ. Next, a *weighting* stage weights these M proposals by the product $q(\mathbf{x}_{i+1}|\mathbf{x}_{i-1,\ldots,i}) \times l(\mathcal{D}(\mathbf{x}_{i+1}))$. The likelihood comprises the image observation model ($\mathcal{C}_2$) and is given by the ratio of two probabilities, for point $\mathbf{x}_{i+1}$ being on or off the boundary. The probability that a point is on the boundary is given by

$$p_{\text{on}}(\mathcal{D}(\mathbf{x}_{i+1})) = p_{\text{on}}(\psi) \propto \mathcal{N}(0, \frac{\sigma_\psi}{|G_I|}), \tag{2.10}$$

where ψ is the angle between the proposed direction and the local boundary direction estimated by

the normal to the image gradient. Equation 2.10 constrains the jetstream to follow the local boundary direction. The spread of p_{on} is modified by the magnitude of the intensity gradient $|G_I|$ so that this constraint is relaxed where the edge strength is weak. The probability that a point is off the boundary is given by

$$p_{\text{off}}(\mathcal{D}(\mathbf{x}_{i+1})) \propto \exp\left[\left(\frac{-|G_I|}{< G_I >}\right)^2\right], \tag{2.11}$$

where $< G_I >$ is the mean gradient magnitude, so that the jetstream prefers to lie on strong edges. Note that the weighting only depends on the proposed point location and the last completed particle section $\mathbf{x}_{i-1,\ldots,i}$, making this a Markov process. The M weights are used as a discrete approximation of the posterior density on φ. The third stage of the algorithm, *importance sampling*, draws samples from this posterior with replacement, to complete one *step* of the boundary tracker.

Jetstreams terminate after a fixed number of steps, referred to hereafter as a *run*. The open contour is then defined as the mean path over the particle set, obtained by taking the mean x and y coordinates at each point.

The jetstream algorithm incorporates an extra procedure for handling sharp corners. A pre-processing stage uses a standard corner detector to identify sharp corners. The algorithm then relaxes the smoothness constraint by replacing the normal prior angular distribution with a uniform distribution at suspected corner pixels. Recently Famao *et al.* [87] improved the corner handling by introducing a variable step length (or 'speed' in the motion tracking analogy).

The fact that jetstream particles progress with Markovian dynamics means that local contour shape is treated as being independent from the whole of a contour. This approach is suitable for ROIs of interest to this project in the absence of a global shape. However, the Markov property means that any jetstream diverging from the desired boundary will fail for subsequent iterations, having no 'memory' of the boundary pixels in its history. Another limitation of particle filter tracking is that the posterior distribution at a given step can be multi-modal, caused by nearby clutter or a *fork* in the tracked boundary. Recently Allen *et al.* [88] presented a similar particle filter for tracking blood vessels in medical images, which handles bifurcations in the forked structures being tracked by a 'rejuvenation' procedure in the Markov Chain.

Another drawback of jetstreams is the lack of a pre-defined termination point for a single run. A user selects the start of the boundary tracker without being able to assert where the tracked section should terminate. Perez *et al.* imply that the boundary tracker should be allowed to diverge from the true boundary, and the user identify the point of divergence. A related drawback is the lack of a global shape model. The algorithm must draw from a static prior angular distribution which gives local smoothness, but discounts any prior knowledge of global shape. In turn, a jetstream is very unlikely to track the whole of a boundary in one run, and so a closed contour is built up in user-defined sections. It is not clear how the final run is expected to reach, and terminate at, the start of the contour. Indeed, examples given in [43] are not closed contours but rather start and end at points on the image border.

2.2.2 Dynamic programming methods

Mortensen *et al* introduced the boundary tracking framework of 'live wires' [89] or 'intelligent scissors' [90], which use dynamic programming as the optimisation scheme ($\mathcal{C}_6$). A live wire is initialised by selecting an anchor on the region boundary. Dynamic programming computes the minimal cost path between the anchor and the moving cursor, where the cost ($\mathcal{C}_4$) is designed to favour short paths that lie along intensity gradients.

After initialisation, the fully interactive framework displays the path *as the cursor moves* allowing the user to 'steer' the contour. When the cursor approximately follows the ROI boundary, a section of the live wire 'snaps' to the boundary in minimising edge energy. At any moment the user can accept a contour section by locating an anchor point to lock the live wire in place, and the process continues with any minimum cost path being that from the cursor to the last anchor point. Performance evaluations in [42] suggest that a live wire leads to higher reproducibility than manual segmentation (98% compared to 96%) although the time taken to segment a region is, on average, twice as long.

Falcão *et al* [42] improve live wires with three key modes of interaction to create the 'live lane' algorithm. First, the user 'trains' the live wire in an initialisation procedure. The operator uses brushes of variable width to define example sections of the region boundary. The program calculates the minimum, maximum, mean and standard deviation of pixel values from these examples. These features are used to update the cost map and the contour subsequently avoids pixels that differ too much from values in the training set. Second, during calculation of the optimal path, the live wire is constrained to exist within a 'lane' that contains the true boundary. This lane is centred on a rough estimate of the ROI boundary as it is traced in real time by the user. Third, when the contour is close to completion, the user constrains the contour to terminate at the starting point with a keyboard interaction. The training step is shown to improve the segmentation of knee bone regions in CT images, where variable edge strength confounds other edge-based segmentation methods. Subsequent experiments in [41] show that an experienced user of live lanes segments talus bones in MR images of the foot with the same accuracy and precision as with original live wires but with up to 31 times the speed.

The main attraction of the live wire method is in its user friendliness. However, live wires have no internal constraints, which can lead to jagged sections in the presence of noise and artificially straight sections between anchors at either side of a weak section [40]. Another limitation of dynamic programming methods is that they give a unique (minimum cost) path between a given pair of anchors, which violates *Lemma 2*. This is in contrast with probabilistic algorithms such as jetstreams, which can in principle produce more than one solution as we will see in chapter 6. If the minimum cost path is not that desired by the user, the only method of control is to place many, close anchors along the boundary, which overrides the algorithm with manual drawing.

Because of these limitations, live wires are often used in conjunction with other interactive tools. In one medical example, Park *et al.* [91] present a protocol for segmenting various anatomical regions in MR images using the live wire, thresholding, region growing and other 2-dimensional, interactive tools in Adobe's 'Photoshop' software. In another medical example, the 'United Snakes' framework

of Liang *et al.* [40] use live-wires to initialise an active contour algorithm. This both speeds up the initialisation process and alleviates problems associated with the sensitivity of snakes to initialisation. Although not necessarily smooth or accurate, the live wire satisfies the requirement that a classical ACM is initialised close to the ROI boundary. The contour then evolves under internal and external energies that introduce smoothness. In addition, a hard constraint forces the final contour to include the initial anchors. The authors segment heart and lungs in X-ray fluoroscopy images, blood vessels, bladder and corpus callosum in MR images, vertebra in X-ray CT images and vessels in an angiogram. Results are compared qualitatively with the use of live wire alone and shown to be superior. The benefits over ACMs alone are in the speed of accurate initialisation. The results above required only around 5 – 10 initialising anchors, and 3 anchors were sufficient to accurately segment the breast region occupying most of a large mammogram (3691×6466 pixels).

2.2.3 Greedy methods

Plummer [22] designed a contouring tool that combines boundary following with local automatic thresholding. The algorithm has never been formally written up (D. Plummer, in correspondence), but is summarised in [92], which compares the tool with thresholding and freehand techniques for MS lesion segmentation. The boundary follower is initialised by a 'seed' pixel located close to, but inside, a lesion boundary. The algorithm initially defines a local point of 'strongest edge' as the largest difference of intensity between any two connected pixels in a square search window centred on the seed. Starting from this point, the algorithm searches four directions ($\pm x$ and $\pm y$) for that with the strongest gradient θ^*. The next boundary pixel is chosen from the four-connected set based upon θ^* and an extra requirement that the corresponding pixel value is above a threshold derived from the seed neighbourhood (it is not clear how the threshold is obtained). The algorithm proceeds by making single-pixel steps around the boundary until the final step reaches the initial boundary point (it is not clear how the steps are guaranteed to terminate at the intended point to close the loop).

Plummer's algorithm frequently fails to extract part of a lesion's boundary and, as a result, freehand delineation is necessary to edit or replace around half of the results [23, 24, 25, 26]. This, and the method's sensitivity to initialisation, could be due to the use of a single seed to initialise both the local search for boundary pixels and the region model.

In a similar boundary tracking framework, Luan *et al.* [93] introduced the 'Filter Function Algorithm'. The filter function is a symmetrical function over angles θ made with the horizontal, in polar coordinates centred on the current boundary pixel. Given an initial anchor on the boundary, the algorithm uses the filter function to track the boundary in the clockwise direction. The algorithm convolves the filter function with 'candidate' pixels that lie on the clockwise side of a radial line passing through the current pixel. By rotating the filter function through discrete orientations, the method seeks both the pixel and the direction that give the maximum response.

As with Plummer's algorithm, it is not clear how the steps are guaranteed to terminate at the starting point to close the loop, the authors merely state that the algorithm 'repeats until the starting pixel becomes a candidate pixel'.

The authors demonstrate the algorithm by tracking the boundary of cranial cross sections in foetal ultrasound images. The results are a visual improvement over the results of gradient filters and an implementation of classical snakes driven by gradient magnitude.

2.3 Modes of Interaction

Recent literature has aimed to improve the efficiency of user interaction rather than removing it. In this section we review the modes of interaction in more detail, in order to highlight their strengths/weaknesses and how they relate to the underlying segmentation algorithm. While we are primarily concerned with boundary-based segmentation, some interactive procedures are more common in region-based methods. We include these interactions for their general applicability to supervised segmentation. In particular, interactions that train a region-based observation model extend naturally to any deformable contour framework driven by a similar model ($\mathcal{C}_2$).

We stated in chapter 1, that balancing requirements of maximal control (*Requirement 1*) and minimal demand (*Requirement 2*) calls for efficient interactions. An efficient interaction maximises the information provided to the algorithm. This means not only providing *more* information, but providing types of information readily used by the underlying algorithms. The remainder of this section discusses interactive procedures in the literature, in terms of efficiency and other merits. We divide the procedures into the categories of *initialisation*, *run-time interactions* and *post editing*.

2.3.1 Initialisation

Initialisation can be region-based or boundary based and can train an observation model or locate the approximate centre or boundary of the ROI. Initialisation can provide both spatial information and image data to train an observation model, which we refer to as the *dual role* of initialisation.

Some initialisation procedures are common to both boundary- and region-based segmentation. A simple example is seeding, where the user locates a pixel or pixels belonging to the ROI, and sometimes the background. Seeding is the only interaction used in classical seeded region growing (SRG), where a single mouse-click provides the location and image intensity to initialise the model [94]. The growing region is an evolving list of connected pixels and the statistics of the corresponding pixel intensities provides the observation model. Instead of a single pixel, seeding can identify a small collection of seeds from a region or background, outlined by the user or marked with a 'brush' stroke. Adams and Bischof [94] suggested that this would provide a more stable model for region growing segmentation, and the approach is commonplace in graph-cut methods [95, 96, 97, 98, 33, 81].

Graph-cut methods are interactively trained upon initialisation by user-defined foreground and background pixels. Following seminal works by Boykov *et al.* [99, 100], interactive graph-cut algorithms have become popular for segmentation in digital photo and video editing [95, 96, 97, 98] and have recently received interest in the biomedical imaging field, for example to segment liver tumours in CT images [13] and neuronal axon regions in electron microscopy images [33]. The method in [13] combines the graph-cut method with watershed and Markov Random Field (MRF) algorithms, but results still rely on interactive post editing by a user-controlled morphological opening operation. The

method in [33] complements the graph-cut algorithm with gradient filtering, but satisfactory results also demand extra interactions in the form of iterative re-training and manual post editing to remove erroneous sub-regions.

Seeding can be region- or boundary-based, reflecting the corresponding segmentation framework and image observation models. Examples of boundary-based seeding are the anchors used in the Live Wire [90] and jetstream [43] algorithms. Classically, these use single pixel locations to initialise the contour model. However, there are examples of boundary-based initialisation with the dual role of training a boundary-based observation model and providing spatial constraint [22, 42]. First hand accounts of using Plummer's algorithm [22] suggest that the dual role can cause adverse sensitivity to initialisation. One regular user of the tool reported great sensitivity, especially for seeds close to the true boundary [19]. This highlights that, while we desire to maximise the information gained from a single interaction, there is a balance between the value of an interaction and how accurate the user input must be. However, despite the need to edit or manually replace around half of the contours, this tool has survived years of routine use at UCL's Institute of Neurology, due in part to the ability for a user to correct erroneous results.

Other than seeding, the literature includes frameworks initialised by placing bounding boxes or initial closed contours. The bounding box initialisation in [33] has the dual role of building a model of the background, and defining a sub-image to be segmented. The spatial constraint of a sub-image greatly reduces the computation time of the associated graph-cut algorithm. Placing an initial closed contour initialises the contour model without training an observation model [5, 9, 77, 78]. The closed contour could be a manual approximation of the ROI boundary or taken from a prior shape model. When no shape model is available, approximate manual initialisation can constrain as well as speed up the evolution [77, 78]. This introduces more demand on the user, which can be reduced by using a second semiautomatic tool to provide the initial contour quickly, as in the use of live wire initialisation in [91, 40].

2.3.2 Run-time interactions

Closed contour models and boundary-tracking methods naturally involve different modes of interaction during run-time. In the case of closed contour models, an intermediate contour can be displayed during evolution, prompting user guidance. Here, 'intermediate' could mean before the termination of an iterative optimisation scheme ($\mathcal{C}_6$) or when the algorithm *has* terminated but the results not yet accepted by the user. Interactions constrain subsequent evolution, either for the remainder of the optimisation scheme, or in a repeated run of the optimisation. An early example is provided by the 'volcanoes' of classical ACMs [5]. Where a deformable contour framework uses a probabilistic objective function, this can incorporate *conditional* probabilities, where the condition is derived from the user interactions. In [56] the authors condition the location of boundary sections in response to interactions.

Boundary tracking methods of live-wires [90] and jetstreams [43] rely on run-time interactions to progress open contour sections around a region boundary. Both methods use anchoring to mark the *last accepted point* after a tracked section veers from the true boundary. In the case of live-wires the

termination of the optimal path at the mouse cursor leads to the real-time *steering* process. Steering has two distinct advantages. First, displaying the optimal path in real-time influences the user's movement of the cursor, and the resulting *feedback* maximises the length of an accepted section. Second, as the path terminates at the cursor, the user can easily create a closed contour from the open-ended live wire. In the case of jetstreams, the particle filtering is not steered in a similar way, but the authors devise run-time interactions that take advantage of the probabilistic nature of the algorithm. Whilst tracking a boundary the user notices or anticipates where image features cause the tracker to take a false path (due to a multimodal posterior as mentioned above). By marking the false paths with a thick line or 'dam', the user assigns zero-probability to any proposal steps that land there.

Another type of user interaction allows the adjustment of parameters between executions of an algorithm. Parameters controlling properties such as contour smoothness can be reset between segmentation attempts (as in [45]) or for different parts of an image. This removes the need for the optimisation of hard-coded parameters and can allow a tool to generalise across multiple applications.

2.3.3 Post editing

Freehand post editing can be used to replace part of a closed contour. However, in this case, contour sections must be both created and removed. This could be done by the successive use of an 'erase' and a drawing tool, but other mechanisms exist. The image analysis package 'MIDAS' (see [36]) realises freehand post editing in the following three steps. First, the user draws manually, along a section of the true boundary missed by the displayed contour (in this case the result of intensity thresholding). This leaves an ambiguous contour comprising two segments that share one boundary section. Second, the user places a marker in any new segment to be included in the region, removing the ambiguity regarding which side of a shared boundary is desired. Finally, the user invokes a 'clean up' procedure to remove the unwanted section and leave a simple closed contour. A similar mechanism is used in the ROI analysis module of the popular 'Analyze' software [101] and in the graph-cut tool recently proposed to segment axonal cross sections in electron microscopy images [33].

Some post editing procedures better exploit characteristics of the underlying contour model. One example is to display the contour as a polygon and allow vertices to be *dragged* to new positions. The authors of [95] report that, overall, users were more satisfied with this interactive tool than with live wires that use run-time steering. The dragging mechanism can be adapted so that boundary points adjacent to that being dragged will also move, maintaining any internal energy constraints. The 'SplineSnake' software in [45] is one example.

In a second example, the authors in [102] provide a novel mode of post editing specific to the underlying segmentation algorithm. Their framework involves a weighted combination of an observation model (image), global shape model and internal constraints of boundary smoothness. Where a section of the contour has missed the true boundary, the user identifies this erroneous region approximately by dragging lines that are displayed perpendicular to the contour, resembling 'error bars'. The observation model is re-weighted according to a Gaussian distribution between these error bars and the boundary section is re-computed. This software is designed for the segmentation of atria in cardiac imaging, but

the general idea of updating a global model in response to post-interaction extends to other applications and models as we will see in chapter 8.

Post editing can also incorporate prior knowledge of the specific application and the type of segmentation error. For example, Heimann *et al.* [103] design efficient post editing for the case where leaking causes a segmented liver region to include the neighbouring kidney in an abdominal CT scan. In this context the authors can be confident that that the leaking originates from the narrowest point between the two regions. The method identifies this point automatically using morphological skeletonisation of the combined regions, and estimates the boundary between the two anatomical regions. Subsequent interaction need only tell the algorithm which of the two regions is the desired (liver) region.

2.4 Notes on Performance Evaluation

Throughout the literature there are varying approaches to performance evaluation of a segmentation framework. This section discusses issues regarding the definition and evaluation of accuracy, and the evaluation of other aspects of a framework's performance.

Accuracy metrics are based on the similarity between a segmentation result and some notion of 'ground truth'. Accuracy is badly defined for the biomedical applications central to the project because of the lack of ground truth. An alternative is to use synthetic images where the ground truth is known. These images must reproduce real-world image conditions if meaningful conclusions are to be drawn from the results. On the other hand, synthetic images allow certain properties of a region or image to be exaggerated and controlled to validate and evaluate the robustness and specific capabilities of a segmentation framework. As well as accuracy, authors evaluate the repeatability of segmentation results. Repeatability metrics are derived from the similarity between two instances of a contour model assumed to segment the same ROI.

Performance metrics that use the similarity between contours must first define what is meant by similarity. Similarity can be computed from a region- or boundary-based definition of the spatial overlap between segmented regions. Common region-based measures are the Dice similarity coefficient [104] and Tanimoto coefficient [105]. For a 'true' region S containing N_S pixels, and its segmentation S containing N_T pixels, these measures are derived from the intersecting region $T \cap S$. For example, the Dice Similarity coefficient is computed by the ratio $\frac{2N_{T\cap S}}{N_T+N_S}$. This project is particularly interested in boundary-based similarity, as we focus on boundary-based segmentation methods and modes of interaction. A popular boundary-based measure is the Hausdorff distance [106]. The Hausdorff distance is a 'maxmin' measure, computed by taking the *minimum* distance from each point on the boundary of S, to the boundary of T, and then taking the *maximum* of these distances over all points on the boundary of S. This measure is sensitive to outlying points of high disagreement and can give misleading evaluations when contours are similar for all but a small section. A popular alternative is to take the *mean* rather than the maximum of the minimum distances [107, 108].

In some cases, overlap measures are replaced by a similarity measure that reflects the application at hand. These use a derived quantity of a segmented region, rather than its spatial properties. Examples include the 'lesion load' relevant to multiple sclerosis studies [92] and the ventricular volume and wall

thickness relevant to cardiology [76].

The literature also reveals qualitative approaches to performance evaluation, to evaluate qualities of an interactive framework such as 'user satisfaction' [109]. This relates to the user's experience, such as how 'easy' or 'frustrating' they found the tool, which is measured using questionnaires. Standard questionnaires such as that in [110] are popular in the wider field of human-computer interaction. However, for systems with the single aim of segmenting ROIs, more specific questions might be appropriate as used in [95].

When comparing one segmentation method with another, the choice of reference method governs what conclusions can be drawn. In many cases, results are compared with manual delineation. The results of manual delineation represent a form of ground truth and, separately, the method itself represents a benchmark. In this study, *Lemma 1* and the implication of *Lemma 2*, that with enough interaction any deformable contour model should give the same result as the user's best manual delineation, justifies the use of manual contours as ground truth when evaluating accuracy. However, for the same reasons, we cannot expect any framework to be more accurate than freehand delineation.

Other authors choose 'state-of-the-art' frameworks to compare with their own. Conclusions drawn from these comparisons are limited as the state-of-the-art is badly defined. A definition based on popularity might reflect a method's age, ease of implementation or user satisfaction rather than the accuracy or chosen performance metric used in experiments. A definition based on the best results quoted in a research paper only holds for the performance evaluation methods and specific data sets used in that paper. Li *et al.* [95] use many performance metrics and test their method on many data sets, but still their evaluation is based on the direct comparison with a somewhat arbitrary reference method.

In order to address the requirement of minimal user demand, (*Requirement 2* in section 1.2), we must also evaluate segmentation methods in terms of demand on the user. The overall segmentation time is not a reliable measure for two reasons. First, a user is likely to use a tool faster as he/she gains experience with it. Second, a user can interrupt a segmentation task before completion, as the framework waits indefinitely for user input. Total time might be used as an indicator of useability in experiments where the user has been instructed to complete the task as quickly as possible as in [42, 41], but this false scenario might compromise segmentation quality. Where two supervised methods share similar modes of interaction it is possible to compare the useability more directly. For example, experiments in [43] compare the demand on the user of two anchor-based interactive boundary-tracking algorithms, in terms of the number of anchors placed by the user.

2.5 Discussion and Conclusions

Interactive segmentation methods in the literature fall short of balancing the requirements of (*1*) maximal user control and (*2*) minimal user demand. The most promising improvements arise from the use of machine learning and the ability to exploit information from the user in an efficient manner. We are motivated to combine machine learning with boundary tracking methods and radial time series.

Boundary-based image models are traditionally based on the magnitude of the image gradient, although some also incorporate gradient direction (eg. [7, 43]). However, for many medical applications

the image gradient is insufficient, and some authors seek more discriminative visual cues including region-based models. In the absence of multispectral data, models based on texture classification give promising results [68, 69, 70]. While some image energies incorporate machine learning, the internal energies of smoothness, stiffness and tension energies do not.

The review motivates the use boundary-based contour models. The Brownian String method taught us that an image model ($\mathcal{C}_2$) and deformation mechanism ($\mathcal{C}_5$) can be particularly suited to, and designed to exploit, the chosen contour representation. In another example, the 1D Cyclic Markov Random Field exploits the radial time series representation. This representation is suitable for star-shaped ROIs, including medical examples in the literature noted in section 2.1.4. Furthermore, we propose that by allowing the user to identify a point near the centre of a ROI, frameworks based on radial time series gain a lot of information from a simple interaction, helping to minimise user demand (*Requirement 2*).

Boundary tracking appeals to our focus on interactive methods. The general scheme of progressing around a region boundary in the arc-length direction (figure2.1) is consistent with the way that a user's fixation follows a boundary during manual delineation. Judging by the popularity of the live-wire algorithm, this results in a user friendly method. Also, it should be possible to make particularly efficient use of user input during segmentation by boundary tracking. However, there is a need for methods that ensure a closed contour.

Where parametric contour models extend to 3 dimensions, methods either replace the 2-dimensional contour representation with a surface mesh [111, 112, 113] or create a 3-dimensional surface by stacking 2-dimensional contours [114, 42, 41, 115, 58, 34]. In the latter case the contour models deform in 2-dimensions only, but information regarding the shape and location of a contour can propagate through successive planes. In principle any 2-dimensional DCM framework can be extended to 3 dimensions by contour stacking but we note that a good DCM should exploit the information between image planes in an efficient manner.

Stochastic optimisation schemes are particularly attractive to the present research for two reasons. The first relates to the prior probability distributions inherent to a stochastic sampling algorithm. These probability distributions are derived from properties of the image or the contour model. Through the use of *conditional* probabilities, we propose that the optimisation can (i) make efficient use of user interactions to condition the observation model, as demonstrated in [56], and (ii) readily incorporate a probabilistic shape models for regularisation. The second attraction to stochastic optimisation stems from their ability to sample from a posterior distribution over contours. The notion of a distribution over plausible results is in keeping with *Lemma 2*, which says that a segmentation algorithm should offer different solutions. We will see in chapter 6, that a distribution over contours lends itself to on-line supervision, whereby alternative solutions are presented to the user for manual selection.

In conclusion, this project will aim to develop DCM frameworks that

- alleviate user demand by incorporating machine learning into one or more of internal energy, global shape models and image observation models,
- seek an image or shape model that will generalise, so that it can be trained on any region-type

having boundary ambiguity or variable shape,

- choose contour representations and optimisation schemes that reflect the other components of a framework,
- use probabilistic optimisation schemes that allow efficient interaction,
- use open contours with on-line supervision based on 'steering' mechanisms,
- use Markovian open contour models for their suitability to unpredictable shapes, but therefore seek methods of closing a contour,
- exploit the simple global constraint offered by polar contour representations,
- seek generalised frameworks that readily exploit information from the user and the image, and potentially from segmentations in neighbouring slices, where the latter allows generalisation for 3-dimensional segmentation by contour propagation,
- use synthetic images in evaluations, for more robuse measures of accuracy, and
- use reference frameworks that have similar modes of interaction to measure relative levels of user demand.

Chapter 3

Statistical Shape Models

Many of the deformable contour models in the previous chapter have been adapted to use machine learning to acquire prior knowledge of the region's shape. Shape priors constrain the segmentation to ensure plausible results and alleviate problems associated with noise and occlusion. They draw from the more general field of shape modelling, which has applications not only in segmentation but also classification, object recognition/categorisation and data compression. Where a model is used in a segmentation framework, this commonly has the role of 'shape regularisation', whereby the objective function ($\mathcal{C}_4$) includes a penalty for contours disagreeing with the prior model (eg. [116, 52, 117]). Shape regularisation is an example of using *discriminative* shape models in segmentation. Another way to introduce shape priors into segmentation is in the use of *generative* models (eg. [118, 119, 120, 121, 122]). In the generative case, a model produces candidate, or *proposal* shapes which share shape properties with the training set. Generating proposal shapes can be viewed as a constrained deformation mechanism ($\mathcal{C}_5$) in a deformable contour framework.

This chapter reviews approaches to shape modelling, with emphasis on methods suited to a supervised segmentation framework. In particular, we are interested in various types of statistical shape model (SSM). We consider as a SSM, any shape model that uses machine learning to give a compact representation of shape information present in training data. The review also focuses on *global* shape models. We use the terms 'global' to refer to information that characterises a whole contour. This definition is not restricted to the case where a boundary has recurring features, but is distinct from the case where local information is simply integrated around a contour (authors such as [11] claim that this leads to a global model). The review looks at SSMs from the point of view of 2-dimensional shape, but some of the methods extend to 3-dimensions.

Two popular SSMs are point distribution models and medial representations. We discuss both of these approaches in order to highlight a fundamental balance between the discriminative power of a SSM and the assumptions it makes about a class of shapes. These SSMs capture high-level information about an object's shape, giving them much discrimination capability, but rely on a high level of similarity between shapes in a given class.

Another family of shape models use the radial time series boundary representation described in the previous chapter. We review radial time series models because they assume relatively little shape simi-

larity, and therefore apply to the types of region considered in this project. The review shows that time series contour representations lend themselves to dynamical modelling techniques, which potentially characterise global shape.

The rest of this chapter is organised as follows. Section 3.1 discusses point distribution models and medial representations, introducing common aims and challenges of statistical shape modelling. Section 3.2 reviews radial time series models, discussing their relevance to the aims of this project, in acquiring prior knowledge of shape and using this in supervised segmentation. Section 3.3 reviews other methods that demonstrate the intention of capturing the most discriminative information with the least complex model.

3.1 SSMs for High-Level Shape Information

This section describes two families of SSM, namely the 'point distribution model' (PDM) and 'medial axis representations' (M-reps). These capture high-level information about global shape characteristics, by virtue of the similarity and points of correspondence between any two shapes of the same semantic class. The PDM and M-reps are both generative SSMs, and use similar methods from information theory. Due to their similarities we review PDMs in more detail and give a shorter overview of M-reps, where the former bares more relation to the current project due to its boundary-based representation.

3.1.1 Point Distribution Models

A Point Distribution Model (PDM) represents a shape as a vector of N points, i.e. coordinates around an object's boundary or, in the case of 3-dimensional models, over its surface [118]. The method is an example of machine learning, where the PDM learns from M training shapes belonging to the class of ROI. The PDM assumes spatial correspondence throughout the training set, between the locations defining each of the N points in a given shape. In general, the correspondence points are distributed evenly between common *landmark* points, which serve to align the training shapes. The training shapes must be co-registered by iterative adjustment of pose (translation, rotation and scaling). The M aligned shapes are stored in the vectors $\mathbf{Y}_0, \mathbf{Y}_1, \ldots \mathbf{Y}_{M-1}$ where the i^{th} shape vector is given by $\mathbf{Y}_i = (x_{i0}, y_{i0}, \cdots, x_{iN-1}, y_{iN-1})^T$. The mean shape and covariance matrix are then given by

$$\bar{\mathbf{Y}} = \frac{1}{M} \sum_{i=0}^{M-1} \mathbf{Y}_i \tag{3.1}$$

and

$$\mathbf{S} = \frac{1}{M-1} \sum_{i=0}^{M-1} (\mathbf{Y}_i - \bar{\mathbf{Y}})(\mathbf{Y}_i - \bar{\mathbf{Y}})^T \tag{3.2}$$

respectively. By Principal Component Analysis (PCA), most of the variability in the training set is represented by the eigenvectors of $\mathbf{S}$ that correspond to the largest eigenvalues. From the $2N$ eigenvectors, the first m are chosen, where $m << 2N$, and stored in the matrix $\mathbf{P} = (p_0, p_1, \cdots, p_{m-1})$. The PDM is then defined by

$$\mathbf{Y} \approx \bar{\mathbf{Y}} + \mathbf{Pa}, \tag{3.3}$$

where $\mathbf{a} = (a_0, a_1, \cdots, a_{m-1})$ is vector of weights, or 'shape parameters'.

The assumption of correspondence leads to two main drawbacks of the PDM approach. First, it assumes that there is correspondence in the first place, between any two shapes in a training set. Considering the simple illustrative example of 'quadrilaterals', the four corners would serve as correspondence points after removing the ambiguity as to which is the 'top left' point and so on. As noted in section 1.1.3, some anatomical regions have recurring features that serve as corresponding points, such as the *spinous process* present on all vertebra (figure 1.2). However, regions such as tumours and lesions can be variable in shape, having arbitrary undulations around the boundary and no sense of 'top' and 'bottom'.

The second drawback of the correspondence assumption is that PDMs require points to be marked on all training shapes by manual or automatic labelling. Manual labelling is labour intensive and not guaranteed to preserve anatomical correspondence. There is active research into automating this procedure, primarily using registration-based methods [123, 124, 125, 126] or 'minimum description length' (MDL) methods [127, 128].

The registration-based method in [123] involves aligning shapes in a training set with their best matching pair, computing a mean shape from each pair and repeating until a single mean shape represents the whole training set. Landmarks placed on the mean shape can then be projected back onto each original shape. This approach, in common with all registration methods, relies on procedures for 'aligning', and measuring the agreement between, pairs of shapes. The authors in [123] use dynamic programming to match boundary sections of high curvature. In [124], the same team replace this alignment procedure with a nonlinear registration method that transforms shapes to minimise the local disagreement between boundaries, defined by non-overlapping areas. Other registration-based approaches in [125, 126] borrow from more familiar medical image registration literature.

Methods of MDL are based on the assumption that the 'simplest' point distribution model is that in which the points $\mathbf{Y}_i$ correspond the most between training shapes. Davies *et al.* [127, 128] borrow from information theory, to define the 'simplest' PDM as that with the minimum 'description length' [129]. Landmark placement then becomes an optimisation problem, where the description length is the objective function and the points $\mathbf{Y}_i$ are first initialised throughout a training set and then manipulated during optimisation. The algorithm requires that each shape is projected into a base domain, eg. a circle in the 2-dimensional case, which causes problems for convex shapes.

At the same time as this project was investigating new shape models (presented in chapters 7 and 8), Berks *et al.* [130] suggested that the MDL method can be used to build a PDM *without* explicit correspondence between shapes. Their method starts with training shapes that have landmarks placed at equal arc-length intervals, starting from a chosen origin. In their example of mammographic masses, there is no obvious choice of starting point, which would correspond between examples. In the absence of a single landmark, the authors choose the optimal starting point for each shape during alignment, being that which gives rise to the minimum variance in the resulting model. The goal of modelling this type of shape (tumour masses) without assuming explicit correspondence is in common with this project. We consider the work of Berks *et al.* to be the closest competitor of methods that we develop in chapters 7 and 8. However, the ultimate aim of [130] is to simulate the shape and appearance of mammographic

masses to provide training and testing data, and more work would be needed to base segmentation algorithms on these models. Also, the evaluation of the method is somewhat limited. The authors use the models to simulate ground truth instances and define a performance metric based on the similarity between generated and original tumours, in terms of combined shape and appearance (texture). Results are compared with an earlier method proposed by the same group [131] which overcame the correspondence problem by enforcing a single landmark point on the tumour mass boundary. The landmark was defined in relation to a nearby, anatomical landmark (the nipple) but there is no confirmed physiological basis for assuming this to be a consistent geometrical relationship.

Other methods of automatic landmark placement draw from the wider geometry and shape literature such as the 'growing neural gas' method [132], 'node splitting and merging' [133] and extracting 'dominant points' from ordered curves [134].

Another limitation of the PDM stems from its linearity (by equation 3.3). The model assumes that any shape belonging to the same semantic class as the training data is given by a linear combination of $m < M$ eigenmodes from the PCA. This in turn assumes that the population of shapes form a unimodal multivariate Gaussian distribution in shape space, and that the training data represent this distribution. According to Cremers *et al.* [135], these assumptions break down when data exhibits complex shape deformations, such as the nonlinear deformations arising from different 2-dimensional cross sections of a 3-dimensional object. The problem is made worse in some medical applications by the presence of pathological variations. The limitations of the linear model have prompted methods that remove its assumptions [136, 137, 138]. An intuitive approach by Cootes and Taylor [137] uses a mixture model to handle multimodal Gaussian distributions. However, this method requires that the number of modes can be estimated. A more recent improvement over the linear model is provided by Cremers *et al.* [135, 138]. Their approach uses kernel PCA, whereby the data are assumed to form a Gaussian distribution in a higher dimensional space, after a nonlinear mapping.

Another problem associated with PDMs is that choices for the number of training shapes and the number of principle modes are somewhat arbitrary. Mei *et al.*[139] showed that this can lead to suboptimal modelling for segmentation, and the task of optimising for the number of distinct modes is the subject of ongoing research [139, 140].

An extension of the PDM adapts it to model multiple regions that are disconnected but part of the same 'constellation' [141, 142]. Examples occur in medical applications where anatomy guarantees a known constellation, such as vertebra making up the spine or metatarsal/metacarpal bones making up joints of the ankle/wrist. The individual regions are modelled by their own mean shape and modes of variation, while the relative positions of the individual regions are assigned a second statistical model, coupled to the first. Constellations do not arise for the applications central to this project, which involve either an individual ROI, or a group (such as multiple sclerosis lesions) that are unpredictable in number and relative position.

3.1.1.1 PDM in interactive segmentation: the Active Shape Model

Cootes *et al.* [9, 118] popularised the use of PDMs for segmentation with the introduction of Active Shape Models (ASMs). The method exploits the generative nature of the model in an optimisation scheme. After initialising with the mean shape placed in an image somewhere near the ROI, the Active Shape model $\mathbf{X} = (x_{i0}, y_{i0}, \cdots, x_{iN-1}, y_{iN-1})^T$ evolves by iteratively varying pose to fit the image and varying shape parameters to minimise the difference between the generated shape and the closest shape $\mathbf{Y}_i$ in the training set. First, the points of $\mathbf{X}$ are subjected to small displacements that reduce an image energy ($\mathcal{C}_2$) to move boundary points toward high gradient magnitude. These displacements are stored in the vector $d\mathbf{X} = (dx_{i0}, dy_{i0}, \cdots, dx_{iN-1}, dy_{iN-1})^T$. Then a least squares estimation calculates the changes in pose, (defined by scaling s, rotation θ and translation of the model centre $\mathbf{X}_c$), that best describes the adjustments $d\mathbf{X}$. The new contour is given by

$$\mathbf{X} = M(s, \theta)[\bar{\mathbf{Y}} + \mathbf{Pa}] + \mathbf{X}_c, \tag{3.4}$$

where $M(s, \theta)$ is a linear transformation for scaling and rotation. To complete an iteration, an adjustment to the shape parameters $d\mathbf{a}$ is calculated by

$$d\mathbf{a} = \mathbf{P}^T d\mathbf{Y}, \tag{3.5}$$

where $d\mathbf{Y}$ is the 'residue' in model space [143], calculated by

$$d\mathbf{Y} = M(s^{-1}(1 + ds)^{-1}, \theta - d\theta)[M(s, \theta)[\mathbf{Y}] + d\mathbf{X} - d\mathbf{X}_c] - \mathbf{X} \tag{3.6}$$

(see [9] and [143] for derivation), where ds, $d\theta$ and $d\mathbf{X}_c$ are the changes in scale, rotation, and translation.

Despite the correspondence assumption of the Point Distribution Model, the impact of the ASM on the medical imaging community has been huge. Examples include the segmentation of ventricles in ultrasound images [144], or bones [145] and the trachea [146] in X-ray CT images to mention but a few. Some refinements to the classical ASM make it even more suited to medical applications. Benefits are seen when the optimisation scheme is replaced with a stochastic method [119], or assisted by user interactions [147, 148, 149].

The interactive algorithm by Hug *et al.* [147] is based on the idea that the most efficient user-guidance comes from interacting with the most influential points on a parametric contour. The most influential points, or 'principal landmarks' are those that carry more shape information, identified by iteratively removing points responsible for the most variation. The remaining points form a coarse 'control polygon' of points that can be placed in an image for fast initialisation. The ASM algorithm is then constrained so that all modes of variation produce shapes that share these control points. The constraint is based on choosing basis vectors that displace the control points in the x- and y-directions, and which have the minimal Mahalanobis norm in the space of principal components.

The 'InterActive Shape Models' (iASM) in [148] also constrain the ASM so that the resulting shapes pass through boundary points defined by the user. Theirs is an iterative scheme, which seeks shapes which occupy the intersection in shape space, between the intersection of the subspace S_1,

spanned by the principal components, and the lower-dimensional subspace S_2 of 'allowed' shapes that contain fixed boundary points. Experiments in [148] suggest reasonable improvements in segmentation accuracy for relatively few interactions, but these experiments use ideal, simulated interactions rather than real user trials. In [149] the same group combine the iASM with an image model exploiting supervised, region-based classification and the hybrid method gives better results in heart segmentation than either of the constituent algorithms.

3.1.2 Medial Representations

Medial representations or 'M-reps' are another family of SSM which, like the PDM, use an explicit parametrisation of object geometry and have advanced the field of segmentation [150, 151, 152, 122]. We give a brief overview of M-reps here to reiterate the assumptions of existing high-level shape models. A detailed account of the method is given in [153].

The main distinction between M-reps and other methods is their use of a medial axis description to parametrise shapes. Parametrisation is based on a 'hub' and 'spoke' model, where points along the medial axis represent hubs, and spokes define straight lines from a hub to the region boundary. Together, the hub and spoke structure is known as an 'atom', whereby the simplest atom comprises two spokes. The number and type of atoms along the medial axis define the full shape representation.

To train M-reps, the atomic configuration is fitted to training shapes, then these are aligned by mechanical deformations. These deformations are modelled by principal geodesic analysis (PGA). Analogous to the PCA above, PGA represents the training set in a space of reduced dimensions, and it is assumed that all objects from the same semantic class as the training set occupy the same space. PGA can be viewed as a generalisation of PCA, where the Euclidean space is replaced by the Reimannian space of medial parameters. As such, the method is theoretically better suited to handle complex modes of variation associated with pathology.

M-reps share two key drawbacks with point distribution models. Analogous to landmark placement for PDMs, the need to assign medial axes and atoms to training data is impractical. This procedure can be partially automated by generating Voronoi diagrams [154, 155], but this in turn must be initialised by boundary points similar to the landmarks of a PDM. Also, as with the PDM, these SSMs assume high levels of shape similarity between regions of interest. Notable successes of the M-rep method are seen for brain structures such as hippocampi and ventricles [156] and the caudate nucleus [152], and for whole organs such as the kidney [156] and liver [157]. However, to our knowledge, M-reps have not been shown to model tumours or other variable shapes as defined in section 1.1.3.

3.2 Radial Time Series models

Section 2.1.4 described deformable contour models that parametrise a contour as a 1-dimensional *radial time series* $\mathbf{r} = \{r_0, \ldots, r_i, \ldots, r_{N-1}\}$. We noted that the $\{\mathbf{r}, \theta\}$ parametrisation limits a contour model to the case of star-shaped ROIs. However, some classes of shape are known to be typically star-shaped and the radial time series re-appears throughout the statistical shape modelling literature as it is convenient and in the case of object recognition, non star-shaped objects can still be classified with

high success rates. Also, some of the techniques apply to the generalised parametrisation $\{\mathbf{r}, \mathbf{s}\}$, where $\mathbf{s} = \{s_0, \ldots, s_i, \ldots, s_{N-1}\}$ are arc-length increments.

3.2.1 Autoregressive models

Kashyap and Chellappa [158] introduced a model using the radial time series representation for the classification and reconstruction of closed boundaries. The authors represent the radial time series by a stochastic model belonging to the family of linear autoregressive (AR) models. In their Circular Aurore-gressive (CAR) model, each point r_i is expressed as a weighted sum of the radii at earlier time points (angles) on the boundary plus a noise term, giving the generalised CAR equation

$$r_i = \alpha + \sum_{j=0}^{m-1} p_j r_{i-j} + \sigma\omega_i, \tag{3.7}$$

where α is proportional to the mean radius, $\mathbf{p} = \{p_0, \ldots, p_j, \ldots, p_{m-1}\}$ is a vector of weights and $m \leq N$ is the number of *lag terms* in the model, also referred to as the *order* of the model. The noise term $\sigma\omega_i$ comes from an independent sequence of normally distributed random variables $\omega = \{\omega_0, \ldots \omega_i \ldots \omega_{N-1}\}$ and standard deviation σ. The full parameter vector for the CAR model is $\mathbf{Q} = \{\alpha, \mathbf{p}, \beta\}$ and is estimated for a given shape using least-squares methods.

In addition to the $\{\mathbf{r}, \theta\}$ parametrisation, Kashyap and Chellappa suggest, without demonstration, that the CAR techniques extend to 2-dimensional series $\mathbf{X} = \{\{x_0, y_0\}, \ldots, \{x_i, y_i\} \ldots \{x_{N-1}, y_{N-1}\}\}$. This *bi-variate* representation extends the CAR model to include non star-shaped boundaries.

Kashyap and Chellappa first suggested that $\mathbf{Q}$ can be used as feature vectors for shape classification. Several authors have subsequently used and refined the CAR model for shape classification in this way [159, 160, 161, 162]. Eom and Park [159] devise a maximum likelihood decision rule classifier to classify outlines of eight aeroplane types and eight machine parts, reporting 100 $\%$ success rate in six out of the eight in each case. Das *et al.* [161] used the 1-dimensional CAR model to classify microbial shape boundaries in classification experiments. The authors report lower success rates between 71.4 $\%$ and 91.4 $\%$ for models of order 2-4. However, these experiments were small, using ten training and five testing images from each class, and used simple 'feature weighting' and 'rotated coordinate system' classifiers. Mir *et al.* [162] use the 1-dimensional CAR model in binary classification to discriminate between liver and kidney boundaries derived from CT images, with 99 $\%$ confidence. The authors also used a bi-variate AR model where points are represented in 2 dimensions as complex numbers, but found no improvement even though some contours were not star-shaped. Although the classification experiments above generally revealed better performance for higher order models, it is noted in [162] that this rule does not always hold and the best model order should be identified empirically for a given application.

Another key extension to the CAR model is given by Dubois and Glanz[163]. The authors change the contour parametrisation to represent non star-shaped boundary as a 1-dimensional series by 'un-wrapping'. First, the boundary points in the $\{\mathbf{r}, \theta\}$ case are stored, where more than one r_i is possible for a given θ_i. The method then steps around the boundary and records the radii ordered by arc-length. However, the resulting 'time' increments become somewhat arbitrary. The arc-length intervals are not

equal and not used. The angles corresponding to the re-ordered points are from an equi-spaced set but the axis is no longer monotonic and has repetitions. The authors acknowledge this loss of 'phase information' and note that a given un-wrapped series does not represent a single unique boundary. The success of the unwrapping algorithm is challenged by experimental results in [164] and [159]. Das *et al.*[164] used two classifiers and two sets of shapes to directly compare the 1-dimensional model with and without unwrapping. Their experiments revealed that the simple 1-dimensional case out-performed the un-wrapping case for all of the tested model orders (1 to 4) on a set of shapes that includes non star-shaped boundaries. Eom and Park's experiments also show that their maximum likelihood classifier using the simple 1-dimensioanl model gives higher classification accuracy, even with non star-shaped boundaries, than the classifier used in [163] after unwrapping. One explanation for the inferior classification rates using unwrapped time series is the loss of boundary phase information [164].

As well as classification Kashyap and Chellappa use the CAR model for encoding shapes to reduce data storage. This relies on the ability to reconstruct a shape from estimates of the parameters $\mathbf{Q}$, which includes simultaneously retrieving the exact noise sequence $\omega = \{\omega_0, \ldots \omega_i \ldots \omega_N\}$ associated with the shape. The authors also note that, by sampling noise sequences from a Gaussian distribution rather than approximating the original sequence, perturbations on the original shapes can be generated. However, this is only used for model validation and to demonstrate the relative information stored in the parameters $\alpha, \mathbf{p}$ and β. The use of CAR as a generative model is not seen in subsequent literature.

Kartikeyan and Sarkar [165] showed that the linear CAR model struggles to classify shapes with complex boundaries and intra-class variability. The authors adapt equation 3.7 to form a nonlinear variant of the CAR model. The so called non-causal quadratic Volterra (NCQV) model is given by

$$r_i = \alpha + \sum_{j=0}^{m-1} p_j r_{i-j} + \sum_{(u,v)\in G} g_{u,v} + \sigma\omega_i, \tag{3.8}$$

where $m \leq N$, $\mathbf{g}$ are the 'Volterra kernels' and G is the set $\{(u,v) : g_{u,v} \neq 0\}$. The procedure to find the volterra kernels and fit the model is complex, eventually giving model parameters $\mathbf{Q} = \{\alpha, \mathbf{p}, \mathbf{g}\}$. The authors use these as feature vectors in a Bayesian schemes for both recognition and classification. Experiments show that the NCQV model improves on the linear case when classifying shapes with more complicated within-class variability.

The NCQV is the only nonlinear radial time series model adapted for shape representation in the literature. The method is complicated and has not received as much interest as the CAR model in later literature. The NCQV is designed for recognition and classification tasks with no obvious extension to shape generation. The authors even state that

> "in the context of closed contour representation, forecasting is not an objective".

In summary, the simple, 1-dimensional, linear autoregressive model has proved to be a useful tool for classification and recognition of star-shaped and non star-shaped boundaries that are not too complex. For the case of non star-shaped objects there is a lack of evidence that classification benefits from the bivariate representation [162] or the use of unwrapping [159, 164]. There is, however, evidence that

introducing nonlinearity into the AR model allows more complex boundaries to be characterised [165]. The CAR model breaks down when used to characterise boundaries that are complex [165], noisy [160] or occluded [166]. There are no examples in the literature, of using CAR models in segmentation, either by developing generative models or adapting the classifiers for shape regularisation.

3.2.2 Markov models

Because the AR approach represents a whole contour by a single set of parameters, He and Kundu [166] claim that it is unable to model 'unpredictable' shapes with 'radical variations'. To address this limitation, and the sensitivity of the CAR model to occlusion, distortion and local perturbation of shapes, the authors combine the autoregressive model with a hidden Markov model (HMM). First, the radial time series of a whole contour is divided into M smaller segments and different AR models are fitted to each segment. The HMM then models the relationship between AR parameter vectors from neighbouring segments. The authors represent a shape as radii separated by equal intervals of arc length rather than angles and a single section of $L < N$ radii $\mathbf{r}' = \{r_1, \ldots, r_l, \ldots, r_L\}$, given the indices $l = \{1 \ldots L\}$, has the AR model

$$r_l = \alpha + \sum_{j=0}^{m-1} p_j r_{l-j} + \sigma\omega_l, \tag{3.9}$$

with parameters $\mathbf{Q} = \{p_0, \ldots, p_j, \ldots, p_{m-1}, \frac{\alpha}{\sigma}, \mu_r\}$, where $\frac{\alpha}{\sigma}$ is a scale invariant ratio of the whole shape's mean radius to the size of fluctuations about the mean and μ_r is the mean radius of the boundary section.

The Hidden Markov Model has M 'states', in this case given by the boundary sections. A state sequence $\mathbf{S}$ of length $T \leq M$ is a vector $\{s_0, \ldots, s_t, \ldots, s_{T-1}\}$ with $s \in \{1, \ldots, M\}$. The relationship between states is modelled by a $M \times M$ transition probability matrix $\mathbf{A}$, with elements

$$\begin{aligned} a_{i,j} &= \Pr(s_{t+1} = j | s_t = i), \qquad i, j = \{1 \ldots M\} \\ &= \frac{\text{N}^o \text{ of occurrences of} \{o_t \in i\} \text{and} \{o_{t+1} \in j\}}{\text{N}^o \text{ of occurrences of} \{o_t \in i\}}. \end{aligned} \tag{3.10}$$

The model requires an initial probability vector $\mathbf{\Pi}$ of length M with elements

$$\begin{aligned} \pi_i &= \Pr(s_0 = i) \\ &= \frac{\text{N}^o \text{ of occurrences of} \{o_0 \in i\}}{\text{N}^o \text{ of training sequences}}. \end{aligned} \tag{3.11}$$

The observation sequence $\mathbf{O}$ has elements o_t given by the vector $\mathbf{Q}$ for the t^{th} segment, and an associated observation density vector $\mathbf{B}$ with elements $b_j(o_t)$ being the *a posteriori* density of observation o_t given $q_t = j$, approximated by

$$b_j(o_t) \propto \exp[-\frac{1}{2}(o_t - \mu_j)\Sigma_j^{-1}(o_t - \mu_j)^T], \tag{3.12}$$

where μ_j and Σ_j are the mean vector and covariance matrix of the j^{th} state calculated by clustering the training vectors. Note that this unsupervised clustering procedure determines the number and length of the state vectors, i.e. the number M of boundary segments.

The machine learning task is to estimate parameters $(\mathbf{A}, \mathbf{\Pi}, \mathbf{B})$, by choosing those that maximise the state optimised likelihood function $p(R, S^*|\mathbf{A}, \mathbf{\Pi}, \mathbf{B}) = \arg\max_S p(R, S^*|\mathbf{A}, \mathbf{\Pi}, \mathbf{B})$. The authors

use the segmental K-means algorithm in [167]. Note that this involves training the model on ground truth shape data to populate the initial probability vector $\mathbf{\Pi}$ and transition probability matrix $\mathbf{A}$ by equations 3.11 and 3.11. Classification then assigns the label p^*, which maximises the likelihood function $p(\cdot|\cdot)$, i.e. $\arg\max_p p(\mathbf{O}, \mathbf{S}^*|\mathbf{A}_p, \mathbf{\Pi}_p, \mathbf{B}_p)$. The authors use the Viterbi algorithm in [168].

The model is translation and scale invariant by virtue of the radial time series representation. However the need for all shapes in a class to be divided into small sections removes the rotational invariance of the model. To account for this the authors attempt to rotate all shapes to a common orientation. This alignment is based on global features of 'elongation axis' and 'minimum radius point' which in turn assumes a certain level of within-class similarity, in conflict with the aims set out in the paper and shared by the present research.

Friedland and Rosenfeld presented another example of using high-level shape information in a 1D-CMRF [73]. The method uses an energy function in the Gibbs formulation that combines low-level and high-level processing. The low-level smoothness term described in section 2.1.4 is complemented with high-level information based on a similarity measure between the contour and the most similar configuration $\mathbf{r}$ in a library. The so-called 'adaptive multi-level energy function' comprises a weighted sum of the low- and high-level energies, where the function is 'adaptive' by varying the relative weights during the optimisation. Arguably, the use of a library in the high-level process means that this shape model is not *statistical*. A statistical model of the population should give a compact representation of the information in the training data rather than explicitly storing that data. The authors demonstrated acceptable results in segmenting objects such as vehicles from infrared remote sensing images, and experiments showed the benefits of the high-level energy over segmentations ran with this term set to zero. However, the library matching process assumes a high level of similarity within the class of shapes, making this approach unsuitable for the goals of this research.

3.3 Other Shape Models

This section reviews Fourier and other shape descriptors that have a role in medical image analysis.

3.3.1 Fourier descriptors

One type of shape model treats a closed contour as a linear combination of sinusoidal functions [169]. A model of order K uniquely defines a given shape with the parameter vector $\{\{a_0, c_0\}, \ldots, \{a_k, c_k\}, \{a_M, c_M\}\}$ used in the equation

$$\begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} a_0 \\ c_0 \end{bmatrix} + \sum_{k=0}^{K-1} \begin{bmatrix} a_k & a_k \\ c_k & c_k \end{bmatrix} \begin{bmatrix} \cos(kt) \\ \sin(kt) \end{bmatrix}, \tag{3.13}$$

where x and y are pixel dimensions and t is an independent parameter related to arc-length.

The Fourier representation has the advantages of controlling the frequency of a shape by the choice of model order, and naturally modelling a closed contour due to the model's periodicity. The model was first used for object classification [170, 171] but also built into a segmentation framework to provide shape priors by Staib and Duncan in [116]. Their framework allowed for machine learning by decomposing training shapes from a given class into the parametrisation above and fitting multivariate Gaussian

statistics to the parameters. The segmentation itself uses a Bayesian approach to find the parametrised shape $\mathbf{Q}$, which maximises the objective function

$$\frac{Pr(\mathbf{D}|\mathbf{Q})Pr(\mathbf{Q})}{Pr(\mathbf{D})} \tag{3.14}$$

where $\mathbf{D}$ is the observation model. In this case $\mathbf{D}$ is a 2-dimensional map of a boundary measure (based on image gradients) so the parametrised contour $\mathbf{Q}$ is first made to conform by turning the 1-dimensional boundary into a 2-dimensional binary image.

Staib and Duncan used the Fourier shape models to segment the Left Ventricle in echocardiogram images and the corpus colossum in MR images. Results seem reasonable and the shape model is intuitive but there were no quantitative conclusions drawn from these demonstrations.

A limitation of the model is the assumption of independence between the model parameters $\{a_k, c_k\}\forall k$. Staib and Duncan suggested extending the method by modelling the covariance of the model parameters . This was later realised by Szekely *et al.* [172], who in turn used the shape priors as regularisers in ACMs to segment the corpus callosum in brain images.

3.3.2 Low-level shape descriptors

Dating back to Hu's 'moment invariants' [173], global geometrical properties such as diameter, aspect ratio or area have been used as low-level shape descriptors. If ROIs in a training set are normalised in terms of scale and orientation, these descriptors can be used as features in shape recognition and classification. Shape descriptors can be region-based or contour-based whereby, as demonstrated in [174], two shapes can be similar according to one and not the other type.

In a recent medical example, Wang *et al.* use shape descriptors to model the shape of ventricles in MRI volumes [10]. Ventricular volume (area) is a well-known indicator of Alzheimer's disease (AD). After choosing the same axial slice from multiple MRI volumes, these are co-registered and the ventricles segmented to produce a binary image. The authors derive two types of novel shape descriptors from a ventricular region and its boundary. The first set of novel shape descriptors are region-based, and designed specifically for ventricle shapes. The 'minimum thickness' is the shortest distance between a point on the left hand side of the region and a point on the right. This always occurs near the centre of the region. Next, the 'axis shape descriptors' rely on landmark points at the four 'corners' of the ventricles. Having identified these points, the authors form descriptors from the two diagonal lengths along with the four distances from the centroid to each corner. The second set of novel shape descriptors are contour-based and are not specific to ventricle shapes. These are derived from the shape's 'signature', which is equivalent to a radial time series. The authors use the mean, variance and higher order moments of the signature as low-level shape descriptors.

The authors also extract standard region-based descriptors of area, circularity, eccentricity, elongation and rectangularity, and the contour-based perimeter. Experiments compare the discriminatory power of the novel descriptors with the standard ones, to recognise AD symptoms. Performance is evaluated by the correlation of a shape descriptor with cognitive test scores (a continuous indicator of AD) as well as binary classification experiments where AD diagnosis provides the ground truth. In correlation exper-

iments the minimum thickness gave the best performance whereas the mean signature value proved the best discriminator in binary classification.

Shape descriptors are generally fast to compute. This makes them suitable for use in real-time applications such as database querying, browsing or recognition tasks applied to video frames [174]. However, shape descriptors are compact and lose a lot of information about global and local shape making them unsuitable for capturing subtle differences between shapes and unable to reconstruct shapes similar to a training set. Also, shape descriptors are not generative because, while values could in principle be sampled at random from learned distributions, a sample would not map to a unique shape.

3.4 Notes on Performance Evaluation

The performance of a shape model is evaluated in two main ways. Where a shape model is used to constrain segmentation, authors can quantify its benefits by comparing the results of segmentation with and without the shape prior (eg. [53]). Irrespective of its use in a segmentation framework, the discriminatory power of a shape model can be evaluated by techniques from the fields of object recognition and classification literature. Recognition aims to determine whether a shape belongs to a given class while binary classification aims to assign a shape to one of two classes. In terms of performance evaluation these aims are closely related as in the recognition case, rejecting a test shape from the single, positive class is equivalent to assigning that shape to an arbitrary negative class in binary classification.

The literature contains different approaches to the choice of a negative class. Where a shape model is used for detection or retrieval, authors often use a database of many shape classes, wherein all but the positive class make up a pooled negative class (eg. [174, 175]). In Wang *et al.* [10], a negative class of shapes is inherent in the chosen application, as the diseased ventricle shapes being modelled are naturally paired with the negative class of healthy ventricles. The binary classification experiments of Mir *et al.* [162] sought to distinguish liver from kidney contours in CT slices. This is not a realistic task as, while these contours might be of similar shape and size in certain axial slices of a tomographic image, they are identified by their anatomical location. However, the liver and kidney classes are a useful complementary pair as the contours possess certain similarities. First, both classes represent deformable organs with similar surface properties. Second, the same image contrast/noise properties govern the quality of ground-truth in both cases.

In the object recognition literature, performance is evaluated by measures such as 'precision' and 'recall'. Adopting the convention that binary classification assigns a 'positive' or a 'negative' label, precision is defined by the true-positive fraction $TP = \frac{N_{\text{TP}}}{N_{\text{TP}}+N_{\text{FN}}}$, where the number of 'true-positives' N_{TP} is the number of test cases correctly assigned to the positive class, and so on. Similarly, 'recall' is given by the fraction of true positives relative to the size of the whole test set.

The work of Wang *et al.* [10] also demonstrates the use of application-specific performance measures. Cognitive test scores provide a secondary measure of the presence or severity of Alzheimer's disease, known to correlate with the change in ventricular appearance captured by the shape models. The test scores are used as a 'ground truth' on a continuous scale and correlation with the variation of shape descriptors provides a quantitative performance measure.

3.5 Discussion and Conclusions

This review suggests that the linearity of the classical PDM may not handle pathological variations, which are the essence of variability in the classes of shape such as lesions and tumours. Also, the assumption of correspondence makes PDMs inappropriate for these applications. However, by exploiting its generative nature, the PDM forms the basis of a popular segmentation framework (ASM) and recent improvements make ASMs more suitable for other medical applications.

Time series models have been used for shape modelling. The key methods can be divided into autoregressive and Markov random field models. The main difference is that autoregressive models characterise the whole of a shape boundary by a set of global parameters whereas the Markovian models model the variation of local shape properties around the boundary. The rest of this discussion highlights the applicability of the radial time series representation to this project (section 3.5.1) and section points out where current shape models that use the radial time series are lacking for our purposes (section 3.5.2).

3.5.1 Radial time series representation

The radial time series allows us to represent multiple shapes in the same parameter space without the need for correspondence between those shapes. The example of polar parametrisation $\{\mathbf{r}, \theta\}$ is limited to star-shaped regions, but medical imaging involves many regions of interest that are star-shaped such as those listed in section 2.1.4. Also, there is evidence that classification is robust against the presence of non star-shaped examples in a training or testing set. The data sets available to this project, of MS lesions and liver tumour cross sections, are both around 80% star-shaped.

We propose that two of the problems associated with boundary following segmentation would be alleviated by the use of radial time series. First, the polar parametrisation can be used to aid loop closing, as a boundary should be completely tracked over a 2π range. Second, because the model contains an estimate of the region's centre, divergence from the true boundary can be avoided by penalising a monotonic increase in radial distance from the centre. The representation may also help to balance the requirements of increased user control (*Requirement 1*) and reduced user involvement (*Requirement 2*). The position of a region's centre, along with an estimate of the mean radius, provide much information to the model while these can be estimated by observations from an image model and/or user interactions.

3.5.2 Radial time series shape models

Autoregressive models are an example of machine learning for shape. These are used for classification but there are no attempts in the literature to develop them for shape regularisation for segmentation. The CAR can be used to generate random perturbations of the single shape it was trained on, but no schemes have been presented for training the CAR on multiple shapes and/or generating contours representative of a given class. The Autoregressive method extends to a nonlinear model as seen in [165]. However, this model is not generative and is only used for shape classification. Kartikeyan's model is also complicated to train, relying on inverse Fourier transforms and nonlinear least squares fitting.

Markov Random Fields in 1-dimension are used for segmentation but do not generally use machine learning. The exceptions are the combined use of a Hidden Markov Model with the AR method [166] and

the use of 'libraries' in the potential function for 1D-CMRF [73]. However, these assume a higher level of similarity within a class of shapes and introduce the same problems of correspondence associated with a PDM. Also, the shape prior based on a library of training series is not strictly a SSM. The Markovian approach extends to open-contour modelling, seen mainly for left ventricular regions, which do not have closed boundaries as the region adjoins the aorta.

In conclusion, global, high-level shape priors benefit segmentation frameworks but existing models are over-constraining for the case of unpredictable, pathologically variable ROIs without spatial correspondence. We are motivated to develop novel shape priors, where the radial time series contour parametrisation offers a promising starting point. Cootes *et al*'s use of the PDM in active shape models also shows that exploiting the generative nature of a SSM in this way can have a huge impact on the field of segmentation. This project will explore the time series approach to develop SSMs that

- capture global shape information without correspondence points,
- are generative,
- model Markovian dynamics in common with boundary tracking frameworks,
- model periodic dynamics associated with closed boundaries,
- are nonlinear,
- use machine learning for global shape information,
- offer discriminative models for shape regularisation,
- are probabilistic, whereby samples drawn from a distribution of likely shapes may form the basis of probabilistic segmentation, and
- allow shape generation to be constrained by observations.

Chapter 4

Machine Learning Classification

'Machine learning' is a general term applied to tasks ranging from parameter estimation to modelling cognitive processes [176]. In the context of a segmentation framework, machine learning is used to provide prior knowledge about the expected shape ($\mathcal{C}_3$) or observation model ($\mathcal{C}_2$). The previous chapter stated that SSMs are examples of machine learning, as the various models estimate parameters from training data. This chapter looks at machine learning techniques appropriate for an image model, which fall into the category of 'supervised classification'. The supervised approach is in line with the intention to make efficient use of information provided by the user. This information comprises ground truth image data that are labelled by human observers. For example, previously labelled data can train the classifier off-line and the results used to provide an improved observation model by pre-processing. Alternatively, an observation model could be trained or refined interactively, using data located in the image during initialisation or run-time. In both cases the data is in the form of feature vectors $\mathbf{x}$, each with an associated class label $y \in \mathcal{Y}$. In the case of binary classification, two labels $\mathcal{Y} = \{1, -1\}$ correspond to positive and negative classes such as 'region' and 'background'.

Machine learning classifiers can be divided into generative and discriminative methods, also known respectively as 'statistical' and 'distribution-free' [177]. In generative methods, feature vectors are related by an underlying joint probability. If we compute the conditional probabilities $\Pr(\mathbf{x}|y), y \in \mathcal{Y}$ and the class probabilities $\Pr(y)$, then we can obtain the probability that unseen data $\mathbf{x}'$ belongs to class y' using Bayes rule

$$\Pr(y'|\mathbf{x}') = \frac{\Pr(\mathbf{x}|y')\Pr(y')}{\Pr(\mathbf{x})}, \tag{4.1}$$

where $\Pr(\mathbf{x}) = \sum_{y \in \mathcal{Y}} \Pr(\mathbf{x}|y)\Pr(y)$. It is then straightforward to classify unseen data using the Bayes classifier. In the binary case we assign label $y = 1$ if $\Pr(1|\mathbf{x}') > 0.5$ and $y = -1$ otherwise.

Generative methods are well principled and require relatively little computation, but are only appropriate for classification problems where we have some prior knowledge about the underlying probability models $\Pr(\mathbf{x}|y), y \in \mathcal{Y}$. Conversely, discriminative methods are favoured when we cannot assume the form of the probability distributions, but can be computationally expensive. In this project, image models will be based on regional texture, for which we have no prior knowledge of underlying processes giving rise to the observed data. Also, we seek classification methods that generalise between applications, and in the presence of high within-class variability for a given application. For these purposes we choose

discriminative methods. These methods, also called 'discriminant analysis', separate data by discriminant functions $g_y(\mathbf{x})$ in the space of the feature vectors. Section 4.1 summarises discriminant analysis and introduces the principle behind nonlinear methods. Section 4.2 gives a more in-depth review of the 'support vector machine', an example of nonlinear discriminant analysis favoured for its ability to generalise. For example, SMVs can discriminate between two classes that might each give rise to multiple clusters in feature space. In the context of segmentation this applies to the classes of 'region' and 'background' textures whereby each class can possess more than one distinct sub-class due to complex textures or within-class variability. The review highlights other benefits of SVMs for use in textured images.

4.1 Discriminant Analysis

Discriminant analysis is a distribution-free approach to classification, which separates data into N-classes by defining $(N-1)$ discriminant functions. In the binary case where $\mathcal{Y} = \{1, -1\}$ we seek the single function $g_y(\mathbf{x})$ whereby

$$y' = \arg\max_{y \in \mathcal{Y}} g_y(\mathbf{x}). \tag{4.2}$$

Equation 4.2 gives rise to regions in feature space $R_1 = \{\mathbf{x} : g_1(\mathbf{x}) > g_{-1}(\mathbf{x})\}$ and $R_{-1} = \{\mathbf{x} : g_{-1}(\mathbf{x}) > g_1(\mathbf{x})\}$ and these regions are separated by a *decision boundary* $\mathbf{x} : g_1(\mathbf{x}) = g_{-1}(\mathbf{x})$.

Discriminant analysis can be divided into linear and nonlinear methods. Linear discriminant analysis (LDA) calculates a discriminant function $g(\mathbf{x})$ from a weighted sum of the components of the d-dimensional feature vector $\mathbf{x} = \{x_0 ... x_{d-1}\}^T$, i.e.

$$g(\mathbf{x}) = \sum_{i=0}^{d-1} w_i x_i. \tag{4.3}$$

where $\mathbf{w} = \{w_0 ... w_{d-1}\}$ is a vector of weights. A hyperplane is defined by constant $g(\mathbf{x})$ in the d-dimensional feature space. The task of classification is to find the hyperplane that optimally separates two classes. After training, unseen data are classified by which side of this decision boundary they lie and the perpendicular distance to the boundary gives a measure of the posterior probability of class affiliation.

Nonlinear discriminant analysis is a family of techniques used when classes are not linearly separable. The general approach is to perform a nonlinear mapping of the data, so that linear techniques can be performed in the new feature space. Hastie *et al* [178] introduced 'flexible discriminant analysis' (FDA), which uses an explicit mapping to a higher dimensional feature space and then uses linear regression to fit a discriminant function in the new space. More recent nonlinear discriminant analysis [179, 180, 181]is based on *implicit* mappings achieved by the kernel method. In these cases $\mathbf{x}$ is replaced by a kernel function $\mathbf{K}(\mathbf{u}, \mathbf{v})$ where $\mathbf{u}$ and $\mathbf{v}$ are any two vectors in the original space. Kernel methods are further extended by the kernel perceptron [182] and the support vector machines described next.

4.2 Support Vector Machines

A Support vector machine (SVM) [183, 184] is a universal classifier that can handle input patterns of high dimensions, and from classes that are not linearly separable. SVMs are characterised by the 'kernel trick' introduced above and the assignment of margins to the decision boundary. A SVM seeks a linear discriminant function in a feature space defined by $\mathbf{K}(\mathbf{u},\mathbf{v}) =< \phi(\mathbf{u}), \phi(\mathbf{v}) >$, where $\mathbf{u}$ and $\mathbf{v}$ are any two vectors in the original space and ϕ is a nonlinear kernel function. The kernel maps the data into a higher dimensional feature space H. Popular choices for the kernel function are polynomial and radial basis functions (RBF). The second characteristic of SVMs is the assignment of margins to the hyperplane. The optimisation maximises the separation of parallel margins that lie either side of the hyperplane.

Support vector machines are trained by casting the problem as a Lagrangian optimisation. A standard result is the 'dual form' of the SVM problem where, for a training set of N examples, optimisation is equivalent to maximising

$$\arg\max_{\alpha} \left(\sum_{i=0}^{N-1} \alpha_i - \frac{1}{2} \sum_{i=o}^{N-1} \sum_{j=0}^{N-1} \alpha_i \alpha_j y_i y_j \mathbf{K}(u_i, u_j) \right), \tag{4.4}$$

subject to the constraints

$$\sum_{i=0}^{N-1} \alpha_i y_i = 0 \qquad \text{and} \qquad 0 \leq \alpha_i \leq c \qquad \forall i, \tag{4.5}$$

where $\mathbf{u}$ are the training vectors, N is the number of training vectors, $\mathbf{y} \in \{-1, 1\}$ are the class labels, $\alpha_i \geq 0$ are Lagrange multipliers and c is a constant revisited later. Solution of equation 4.4 is costly, as the size of $\mathbf{K}$ is equal to the square of the number of training examples. Formally, this is a Quadratic Programming (QP) problem to optimise a convex quadratic objective. There are different algorithms available for solving this type of problem [185, 186, 187, 188], which all involve breaking them into smaller QP problems.

After optimisation, any input vector u_i for which the corresponding Lagrange multiplier α_i is greater than zero, is a support vector and will be denoted $\mathbf{S}_i$. These vectors lie on the maximal margins, parallel to the hyperplane. The margin maximisation may allow some vectors to lie *between* the margins, having a closer distance to the hyperplane than the support vectors. These so called 'slack variables' are assigned a cost governed by their distance from the hyperplane and weighted by the constant c. This constant therefore controls a trade-off between separating the margins and keeping all training data outside them.

A trained SVM is defined by the support vectors $\mathbf{S}$, a vector of signed Lagrange multipliers $\alpha\mathbf{y}$ and a 'bias' b which gives the perpendicular distance from the hyperplane to the origin of H. This offset enables hyperplane margins to reside in positive and negative subspaces. A new observation has an associated feature vector $\mathbf{u}$, and is classified by evaluating

$$d_{\text{SVM}} = \sum_{i=0}^{N_s-1} \alpha y_i K(\mathbf{u}, \mathbf{S}_i) + b, \tag{4.6}$$

where N_s is the number of support vectors and d_{SVM} is the distance to the hyperplane, or 'decision value'. The sign of d_{SVM} gives the label of the predicted class and magnitude of d_{SVM} is a relative measure of the certainty of class membership.

One intuition behind SVMs is that by maximising the hyperplane margins we obtain good generalisation. It may be the case, however, that the dimensionality of the input feature space is high enough to compromise generality. Yao *et al* [189] address this problem by splitting the feature vectors into sub vectors, each the subject of a separate SVM. A 'voting committee' of SVMs is then defined and can be optimised for the particular configuration of how the features are split up.

Due to the construction of a single decision plane in feature space, SVMs are naturally a binary classifier. However, their generality and computational efficiency has motivated the development of new SVMs for classifying arbitrary numbers of mutually exclusive classes (see e.g. [190, 191, 192, 193, 194, 195]).

One limitation of SVMs is that the output is an un-calibrated, unbounded value $d_{\text{SVM}} \in (-\infty, \infty)$. This limits the generality of the distance measure. Ideally we would like to calibrate decision values to give estimates of the posterior probability of belonging to each class. Calibrating SVM outputs is non-trivial because typical distributions of decision values d_+ and d_- are badly behaved. Platt [196] showed that bounded probabilities can be approximated by fitting a parametric sigmoid function P_{sig} between the limits d_{min} and d_{max}. After obtaining a decision function $f(u_i)$, the probability that a new feature vector u_i has label $y_i = 1$ is approximated by

$$\Pr(y_i = 1|(u)_i) \approx P_{\text{sig}}((u)_i) \equiv \frac{1}{1 + \exp[Af((u)_i) + B]} \tag{4.7}$$

where A and B are the parameters sought in fitting. To obtain A and B the training data must be partitioned and used in an iterative training and testing algorithm, which is computationally demanding.

Another drawback of SVMs is that parameters must often be optimised for the kernel function. Hsu and Lin [197] point out that in particular, the influence of RBF parameters γ and c are inter-dependent so we seek an optimal *pair* of these parameters. The authors use a grid-search where parameters are varied exponentially and the 'fitness' observed at discrete intervals. The fitness is related to the accuracy of the corresponding SVM, which must be estimated. We discuss methods of performance evaluation in section 4.3.

4.2.1 Featureless classification for texture

This section reviews the use of SVMs for featureless texture classification. These methods exploit the fact that a single texture in an image is completely described by the relative intensity and relative positions of pixels. These two pieces of information are encoded in any vector of two or more pixel values taken from known relative locations. The methods described here use such vectors as inputs in a SVM. The kernel trick then implicitly extracts the features that best discriminate the classes.

The earliest example is found in the work by Kim *et al* for the classification of text regions in video frames [198, 199]. The authors use vectors of image intensity taken from a sampling window, to form patterns associated with the pixel at the centre of the window. The sampling window is a star-like

configuration of pixel locations. This window shape was introduced by Mao and Jain [200] in their 'simultaneous auto-regressive' (SAR) texture model, where multi-scale texture features are captured by associating different model parameters with different neighbourhood sets at increasing radial distances from the central pixel. The resulting SVM is compared with a Neural Network (NN) classifier in [201]. With the same input patterns, the SVM out-performs the NN method, having a 91.2% success rate compared with 86.3%. This suggests that the combination of an autoregressive sampling window and the SVM's kernel trick provides a powerful texture classifier without explicit feature classification. However, the SVM required twice as much processing time as the NN classifier in this experiment.

The featureless SVM texture classifier is virtually invariant to rotation, if the training set is large enough to involve a given texture at all practical (discrete) orientations. Scale invariance cannot be expected with a finite window size. However, larger scale textures are captured in part by any given window, so by the same reasoning as for rotation invariance, a large enough training sample could lead to virtual scale invariance. Campanini *et al* [202] have developed a scale invariant method for SVM texture classification without features by using multiple sampling windows of different dimensions. In order to keep the dimensions of the input patterns constant, the larger scale windows are sub-sampled. Campanini's method is used to classify the whole of a mammogram image as 'suspect' or 'not' based on the likelihood of lesions at any of the feasible scales.

Our research group have previously used raw pixel values as input dimensions for 2-dimensional texture classification in medical images [203, 204] and an extension to 3-dimensions was investigated during the early stages of this project [205].

The technique is further extended to classify 4D functional MRI data by Mourao-Miranda *et. al* [206]. The authors are concerned with the task of classifying 4D datasets, comprising time sequences of 3-dimensional brain volumes. Patients perform tasks during image acquisition, so that functional information can be derived from the correlation between fluctuations in a blood-oxygen level dependent signal (BOLD response) and the temporal windows in the task's design. Datasets can then be divided according to which task is performed or, in the case of Alzheimers studies, whether the participant has the disease or not. At the time of their study, Mourao-Miranda *et. al* note that machine learning techniques had only been applied to such tasks by two previous studies in 2003. In both cases, fMRI sequences were characterised using feature selection methods. The novelty in [206] comes from their use of voxel indices as input dimensions in a hyperspace classifier, in much the same way as described for texture above. The whole of a 3-dimensional volume defines an input feature space of hundreds of thousands of dimensions, (in practice the authors reduce this to hundreds of dimensions by PCA).

4.2.2 On-line algorithms and incremental learning

The classical SVM algorithms mentioned above are examples of off-line or 'batch' algorithms, requiring all of the training data upon initialisation. On-line algorithms on the other hand inspect the training examples sequentially. Consider the general on-line case of a discriminant algorithm that iteratively updates a decision boundary at time t upon inspection of training data $\mathbf{u}_t \cup \mathbf{u}_{t-1}$. This could be at one iteration during the inspection of a fixed set of training data, or upon revision of a pre-trained classi-

fier using new training data. The second case refers to hyperplane 'tracking' or 'incremental learning' necessary in cases where training data varies over time.

Recently, on-line algorithms have been proposed for training hyperplane classifiers of the SVM type, having maximal-margins and kernel space mapping. In these cases, support vectors are sequentially added to the set defining the hyperplane margins. When new support vectors are added it is desirable for some 'old' ones to be removed for three main reasons. First, some of the existing support vectors may have been erroneously included due to the lack of training data previously available. Second, in some scenarios, the newer data may be expected to better represent the set to be classified, as it is part of the same local or temporal subset. Finally, the removal of support vectors becomes necessary to avoid an unbounded increase in memory storage and classification time associated with a trained machine.

Crammer *et al* [207] introduce a 'fixed budget perceptron' algorithm that seeks to remove the most redundant support vectors as more are added. The method simulates the removal of each support vector to identify that which, when removed, remains correctly classified with the largest margin. Removing this vector fixes the number of support vectors and leaves the hyperplane $\mathbf{w}_t$ no more complex than at time $t-1$. The method is shown in [208] to perform well on relatively noiseless problems but degrade quickly with increasing noise. Weston *et al* [208] introduce the 'tighter budget perceptron', which uses a new method for selecting which vectors to remove. The authors replace Crammer's simulation with a direct evaluation of the misclassification rate. The tighter budget perceptron is shown to give significantly better classification rates than Crammer's algorithm as the budget of support vectors increases, until a limit is reached above which they appear to perform equally. More recent refinements to fixed budget on-line learning involve removing the least recently included support vectors, as in the 'Forgetron' of Dekel *et al.* [209] and even removing support vectors at random [210].

There is strong motivation for on-line algorithms, as batch training will fail in the case where training data are either non-stationary [211, 212] or large [213, 214]. However, while there are well accepted implementations of *batch* learning ([215, 216, 187]), implementations of on-line algorithms are scarce and hardly used in the machine learning community. This reflects practical difficulties regarding memory allocation and data types [217].

4.3 Performance evaluation

Methods of performance evaluation are affected by the type of classification task and in some cases the classifier itself. In an example of the latter, it was originally thought [184] that the number of support vectors N_s, as a proportion of the number of training data, is an upper bound on the classification accuracy that a trained SVM can achieve. However, Barzilay and Brailovsky [218] show that a performance measure based on N_s is not always applicable. Instead, an empirical performance measure can be used to assess classification accuracy. These use labelled testing data and return a measure based on the proportion of correct classifications and/or misclassification.

4.3.1 Receiver Operating Characteristics

A method of classifier evaluation popular in the medical image analysis community is receiver operating characteristic (ROC) analysis (see eg. [219] and Appendix C in [220] for an overview). ROC analysis extends the measure of 'precision' described in section 3.4, for cases where a classifier gives a continuous measure of class membership, such as the probability returned by LDA or the decision value returned by a SVM. After assigning values to all test data, a unique binary classification is defined by thresholding this value. Each classification (threshold) has associated true positive fraction TP and false positive fraction FP given by

$$TP = \frac{N_{\text{TP}}}{N_{\text{TP}} + N_{\text{FN}}} \qquad \text{and} \qquad FP = \frac{N_{\text{FP}}}{N_{\text{FP}} + N_{\text{TN}}}, \tag{4.8}$$

where N_{TP} is the number of true positives etc. as introduced in section 3.4. Varying the threshold from the minimum (negative) decision value to the maximum (positive) yields pairs of TP and FP. The final stage is to construct a ROC curve by plotting, FP against TP.

The area under a ROC curve (AUC) has a value between 0 and 1, where 1 indicates perfect classification, 0 would mean that all data are wrongly classified and 0.5 corresponds to discrimination capability no better than random label assignment. Swets [221] showed that, if the distance value assigned to each class a and b follows a normal distributions with means μ_a, μ_b and standard deviations σ_a, σ_b respectively, then the AUC is related to the separation of these distributions by the inverse cumulative normal distribution function.

4.3.2 Cross-validation

Cross-validation is a general term for methods that evaluate classifier performance, which are different for 'one-class' and binary classification.

In the case of one-class classification, a trained classifier is tested by assigning scores or labels to data in the test set. As the test set is known to belong to the class, high scores, or an abundance of positive labels, indicates a successful classifier. However, a single pair of training and testing data sets can give biased results. Cross-validation partitions the data into multiple training and testing sets, and repeats the train/test procedure, then the mean of the chosen performance measure (eg. AUC) is less prone to bias. However, the choice of data partition is not always obvious. The size of the test set introduces a trade-off between bias and the variance in performance measures. One option is to omit each feature vector in turn from the training set. This *leave-one-out* procedure must be repeated N times where N is the size of the ground truth set. Another option is to use *k-fold* cross-validation, which is flexible in terms of the number k of training/testing sets and how they are combined in a single classification. Medical imaging studies may present natural sub-sets, which give meaningful partitions. For example, data can be divided by patient or time point in a longitudinal study. This applies to both one-class classification and the binary case, where a classifier trained on positive and negative data from one or more patients is used to classify data from another.

4.4 Discussion and Conclusions

SVM classifiers are relevant to the present research for three main reasons. First, SVMs are able to handle input feature spaces of high dimensions, which might be necessary when classes such as textures can not be discriminated by low-dimensional information. Second, the maximal margin method leads to good generalisation capacity ([202, 222]), which is important for applications suffering inter-class variability. Third, the kernel trick allows a simple method of texture classification discussed in section 4.2.1, which extracts features implicitly from the relative location and intensity of neighbourhood pixels. This approach is attractive to the present research for its generality.

Support vector machines offer a powerful classification tool which, when tuned for a certain application, should generalise well in the presence of within-class variability. There is evidence that the kernel trick, along with a well chosen sampling window, enables image texture to be classified well without explicit feature classification. This kind of universal texture classifier is attractive to a medical image analysis package because (i) texture is a key visual cue in low contrast monochromatic data, (ii) regions of interest such as those associated with disease often vary in appearance for different patients, time points or image acquisitions, and (iii) the same method should generalise well across different applications in a common package, since the kernel trick implicitly extracts texture features from raw data without application-specific pre-processing.

A common criticism of SVMs is that they are limited to binary classification. This is not considered a limitation for the present research, where a single (positive) class represents the region of interest.

There is motivation for the implementation of incremental learning algorithms, both to overcome large demands on processing and memory storage, and to enable classifiers to evolve according to changing or growing sets of training data. The idea of improving or localising a classifier in the light of new ground truth is attractive to a supervised image analysis package where many regions belonging to a certain class might be segmented in turn. In the case of medical image segmentation, this extends to cases where (i) ROIs (or their cross section) are segmented on successive slices in a 3-dimensional dataset, and (ii) many examples of ground truth from numerous imaging centres may become available, given the increasing acceptance of medical image databases and standardisation of image archiving.

In conclusion, this project will draw from the machine learning classification literature in order to

- create image models based on supervised binary classification of regional texture,
- use featureless texture classification, in order to generalise for applications with large within-class variation and for different applications,
- use these image models to disambiguate region boundaries for interactive deformable contour models, with the aim of reducing demand on the user, and
- design the classifiers so that they can exploit new training information provided interactively, with the aim of making efficient use of interactions.

Chapter 5

Nonlinear Time Series Models

In chapter 2 we saw deformable contour models that use a 1-dimensional *radial time series* as a contour parametrisation ($\mathcal{C}_1$). Section 3.2 further described the ways in which time series methods have been used along with the 1-dimensional paramerisation to create statistical shape models for use in object recognition and classification, and where these are in turn used as shape regularisers for segmentation ($\mathcal{C}_3$). This motivates us to look further into the field of nonlinear dynamics, in order to extend the idea of time series modelling for shape. Dynamical models can describe any data that is represented as an ordered series. Examples include physiological time series [223], spatial data including surface models in microscopy [224] and texture features derived from ultrasound time series [225]. We review time series analysis in the proposed context of shape modelling for segmentation frameworks.

Section 5.1 describes Langevin models, which are used to model Markovian dynamics. We view Langevin models as an extension to the 1D-CMRF model and later develop them for shape modelling and segmentation in chapters 7 and 8. Section 5.2 describes Gaussian processes, of which the 1-dimensional case is an example of non-Markovian time series modelling. We view Gaussian processes as an extension to the CAR model and also develop them in chapters 7 and 8. Chapter 7 will introduce key elements of new shape models and devise methods of training a model and 'scoring' unseen shapes, while chapter 8 builds the models into segmentation frameworks, including novel use of generative models. Throughout this chapter we are looking for where Langevin and Gaussian process models might enable us to

- learn from a set of contours, prior shape information at a higher 'level' than local energies,
- work without the assumption of spatial correspondence between shapes in a class,
- derive a 'score' for use in shape regularisation, which estimates the probability that an unseen shape belongs to the same class of shapes as the training set,
- exploit generative models in a probabilistic segmentation framework, and
- adapt generative models to incorporate observations, from information available in the image and/or from user interactions.

5.1 Langevin Models

A Langevin model describes the dynamics of a time dependent state vector $\chi(t)$ as a stochastic process. Langevin models are characterised by a deterministic term $a(\chi(t))$ and a stochastic term $b(\chi(t))$ in the generalised Langevin equation

$$\frac{\mathrm{d}\chi}{\mathrm{d}t} = a(\chi(t)) + b(\chi(t))\omega(t), \tag{5.1}$$

where $\omega(t)$ is uncorrelated, time dependent noise with zero expectation value.

Langevin models assume a Markov property defined by

$$\chi(t) = f(\chi(t - \Delta t)) \tag{5.2}$$

where f denotes any function and Δt is a constant delay parameter for which the Markov property holds. The transition density $\Pr(\chi(t)|\chi(t - \Delta t))$ evolves according to the Kolmogorov forward equation (an example of a Fokker-Planck equation) of the form

$$\frac{\partial}{\partial t}\Pr(\chi(t)|\chi(t-\Delta t)) = \left[-\sum_i \frac{\partial}{\partial \chi_i}(D_i^{(1)}(\chi(t),t)) + \frac{1}{2}\sum_{i,j}\frac{\partial^2}{\partial\chi_i\partial\chi_j}D_{i,j}^{(2)}(\chi(t),t)\right]\Pr(\chi(t)|\chi(t-\Delta t)), \tag{5.3}$$

where the *drift function* $\mathbf{D}^{(1)}$ corresponds to the deterministic term and the *diffusion matrix* $\mathbf{D}^{(2)}$ corresponds to the stochastic term. These are equivalent to the time-stationary conditional moments at position χ in state space given by

$$D^{(k)}(\chi) = \lim_{\Delta t\to 0}\frac{1}{\Delta t}\big\langle[\chi(t+\Delta t) - \chi(t)]^k|\chi(t) = \chi\big\rangle. \tag{5.4}$$

Let us treat a series $\mathbf{x} = \{x_0, \ldots, x_i, \ldots, x_{N-1}\}$ as the time evolution of a 1-dimensional state variable $x(t)$. Let us also define corresponding time intervals $\mathbf{t} = \{t_0, \ldots, t_i, \ldots, t_{N-1}\}$ and assume that the delay parameter can be quantized as $\Delta t = n \times \mathrm{d}t$, where n is a positive integer and $\mathrm{d}t$ is constant. In the following derivations $x(t)$ corresponds to x_i and $x(t \pm \Delta t)$ correspond to $x_{i\pm n}$ and $x_{i\pm 1}$ is the special case where Δt is chosen so that the Markov property holds between successive time points in $\mathbf{t}$.

Langevin models approximate the evolution of $\mathbf{x}(t)$ as a difference equation by adopting Itô's interpretation [226, 227] of a stochastic differential equation (SDE). The time evolution, derived from the Kolmogorov forward equation 5.3, has the form

$$\mathrm{d}x(t) = a(x(t))\mathrm{d}t + b(x(t))\omega(t). \tag{5.5}$$

Assuming stationary dynamics, the deterministic and stochastic terms are not explicitly time dependent, and are given by functions of the state variable $a(x)$ and $b(x)$. These are related to the drift function $D^{(1)}(x)$ and diffusion function $D^{(2)}(x)$ in the underlying Fokker-Planck equation by [228]

$$\begin{aligned} a(x(t)) &= \frac{D^{(1)}(x)}{\Delta t} \\ b(x(t)) &= \frac{D^{(2)}(x)}{\sqrt{\Delta t}}. \end{aligned} \tag{5.6}$$

In the context of radial time series modelling for shape, we wish to 'score' unseen shapes according to their agreement with a model. We also stated that probabilistic scores are preferable, for use in

probabilistic optimisation schemes. In section 7.2.3 we will see that shape scoring can take advantage of the conditional probabilities (transition densities) central to Langevin models. For this we notice that the joint probability of a series of N points under a Langevin model is given by

$$\Pr(\mathbf{x}) = \Pr(\mathbf{x}_0) \prod_{i=0}^{N-1} \Pr(\mathbf{x}(t+\Delta t)|\mathbf{x}_{i-1} = \mathbf{x}(t)) \tag{5.7}$$

and use equation 5.7 in an objective function ($\mathcal{C}_4$).

5.1.1 Parameter estimation

Given a (1-dimensional) time series, fitting a Langevin model involves estimating the form and parameters of the functions $a(x)$ and $b(x)$. This means learning $D^{(1)}(x)$ and $D^{(2)}(x)$ in equation 5.4, for a chosen delay parameter Δt, from observed data.

Friedrich and Peinke [229] introduced the 'direct estimation' method to extract drift and diffusion functions from observed series. The method assumes Gaussian statistics for the transition density $\Pr(x(t+\Delta t)|x(t))$. The authors approximate $\Pr(x(t+\Delta t)|x(t))$ as a function of the state variable in the following steps.

Step 1. Divide the state space into bins of equal width Δx, centred on discrete values x_n.

Step 2. For a given bin, note all observations $x(t)$ that fall in the range $x(t) \in x_n \pm \frac{\Delta x}{2}$.

Step 3. Starting from these observations, follow the time series along a trajectory of length Δt and form a histogram of the future position of the state variable.

Step 4. Use this histogram to approximate the transition density as $\Pr_n(x(t+\Delta t)|x(t) \in x_n \pm \frac{\Delta x}{2})$

Step 5. Estimate the Gaussian mean μ_n and standard deviation σ_n from the distribution $\Pr(x(t+\Delta t)|x(t) \in x_n \pm \frac{\Delta x}{2}) = \mathcal{N}(\mu_n, \sigma_n)$.

Finally, the drift function and diffusion function at x_n, for a given delay parameter Δt, are related to the first and second order statistics of the transition density by

$$\begin{aligned} D_n^{(1)}(x_n, \Delta t) &= (\mu_n - x_n)\Delta t \qquad \text{and} \\ D_n^{(2)}(x_n, \Delta t) &= \sigma_n \sqrt{\Delta t} \qquad x_n \in \{x_{\text{min}}, \ldots, x_n, \ldots, x_{\text{max}}\}. \end{aligned} \tag{5.8}$$

Repeating for all x_n within the state space occupied by the series, and evaluating 5.6, yields discrete approximations of $a(x(t))$ and $b(x(t))$ in equation 5.5. The remaining task is to fit functions to these approximations. It is common to assume a simple parametric function and use a standard fitting procedure to estimate the parameters. Function types are chosen by inspection and might call upon some intuition regarding the physical process underlying the series data.

The choice of Δt depends on the nature of the continuous time process and how the data are discretised. The Markov property is implicit in the two-step conditional probability $\Pr(x(t+\Delta t)|x(t))$,

but for this property to hold, the delay parameter Δt must correspond to that in equation 5.2. This is the characteristic time scale of the Markov process, which does not necessarily correspond to a single time interval separating observations and must be estimated. Over-Sampled data may necessitate a trajectory length Δt of multiple data points to satisfy the model. Conversely, too sparse data might never capture the Markovian property of the underlying stochastic process. It is common to find a delay parameter empirically at the same time as extracting $a(x)$ and $b(x)$ [230, 231, 232, 223]. Other methods for choosing the best delay parameter are given in [233, 224]. Friedrich *et al.* [233] define the optimal Δt as that for which the multiple conditional probability density $\Pr(x(t+\Delta t)|x(t), x(t-\Delta t), x(t-2\Delta t)\ldots)$ and the one-step density $\Pr(x(t+\Delta t)|x(t))$ agrees for all $x(t+\Delta t)$. However, the authors do not elaborate on how these should be evaluated or their agreement defined.

The direct estimation method is a generalised and intuitive way of training Langevin models from series data. Moreover, the method can be extended to learn from multiple time series comprising different realisations of the same underlying dynamical system. In the case of radial time series we show in section 7.2.2 that this is equivalent to training a statistical shape model on multiple shapes of the same class.

Alternative methods of training Langevin models from series data combine maximum likelihood estimation with simulation techniques [234, 235, 236]. The general idea uses the joint log likelihood derived from equation 5.7, given by

$$\begin{aligned}\mathcal{L}(\{x_0,\ldots,x_i,\ldots x_{N-1}\}|\mathbf{a}) &\sim \prod_{i=0}^{N-1}\Pr(x(t+\Delta t)|x_i=x(t),\mathbf{a}) \\ &\sim \sum_{i=0}^{N-1}\log\Pr(x(t+\Delta t)|x_i=x(t),\mathbf{a}),\end{aligned} \tag{5.9}$$

where $\mathbf{a}$ is the vector of parameters of the Langevin equation. The estimation uses an iterative optimisation strategy to find the parameters $\mathbf{a}$ that maximize $\mathcal{L}(\{x_0,\ldots,x_i,\ldots x_{N-1}\}|\mathbf{a})$. The conditional probabilities $\Pr(x_{i+1}|x_i,\mathbf{a})$ in equation 5.7 must be estimated for each interval $[i, i+1]$. All examples in [234, 235, 236] extract these estimates from simulated data, generated using the current estimate of $\mathbf{a}$ and simulation techniques described in the next section. A histogramming procedure similar to steps 2-5 above then yields estimates of the conditional probability density. However, these estimates are likely to be biased and as such the benefits of these maximum likelihood methods is not clear [235, 237]. One variant uses an auxiliary model to approximate the true likelihood [238]. However, the algorithm is complex and a suitable auxiliary model may not be available in all cases.

5.1.2 Simulation

Simulation plays several roles in time series modelling. For example, section 5.1.1 noted that a simulated series can be re-analysed to evaluate $\Pr(x_{i+1}|x_i,\mathbf{a})$ during maximum likelihood parameter estimation. Also, all the examples of using Langevin models given later in section 5.1.4 simulated the dynamics learned from data for validation purposes. Similarly, in our case, simulation allows us to overcome problems associated with the dynamic degradation of radial time series sampled on a pixel grid (section 8.4.1.2). Moreover in section 8.4, we realise novel segmentation frameworks by simulating radial

time series. This follows the intuition that simulation is equivalent to generating hypotheses in shape space, which forms the basis of a stochastic optimisation scheme ($\mathcal{C}_6$).

Following Itô's interpretation [226], a Langevin series considers a stochastic differential to be the limit of a discrete time process. It follows that an instance of a Langevin time series can be generated by the solution of the SDE in equation 5.5. Examples throughout the literature perform numerical integration using the Euler-Maruyama representation

$$x(t + \mathrm{d}t) = x(t) + \mathrm{d}t \times a(x(t)) + \sqrt{\mathrm{d}t} \times b(x(t))\omega(t), \tag{5.10}$$

where $\mathrm{d}t$ is an integration time step.

There are three points to note from equation 5.10. First, the diffusion term scales as the square root of $\mathrm{d}t$. This is necessary to obtain the limit of the diffusion process in the underlying Fokker-Planck equation [239]. Second, for any choice of integration time step $\mathrm{d}t$, the Markovian property 5.2 holds, where $\mathrm{d}t$ is equivalent to the delay parameter Δt. Third, the integration time step $\mathrm{d}t$ has arbitrary units. When simulated data represents a real time series the intervals can be re-scaled to the characteristic time scale. In practice, large values of $\mathrm{d}t$ used in equation 5.10 cause the simulated series to diverge. In general $\mathrm{d}t$ should be just small enough that the conditional density estimates do not change for smaller values [234]. However, there is no single optimal choice that applies to all drift and diffusion functions.

5.1.3 Incorporating observations

We wish to constrain the simulation scheme so that an instance (time series) agrees with both the dynamics encoded in the SDE and observations in state space. For the purpose of this project, 'observation' refers to information from an image model and/or information provided by the user of an interactive segmentation algorithm. In particular, in chapter 8, we will see how Langevin simulations can be constrained so that shapes generated as radial time series agree with both the global dynamical model and evidence of region boundaries in an image.

The goal of constraining generative Langevin models in this way is the subject of ongoing research, particularly in the meteorological literature, where it is known as *data assimilation* [240, 241]. The state of the art generally employs the SDE as a sampling mechanism combined with another inference scheme such as Bayesian MCMC [240, 241] or Gaussian process [242]. We propose that, in the context of radial time series modelling for shape, techniques based on data assimilation can constrain the SDE simulation method above by observations in the form of an image model ($\mathcal{C}_2$) or interactions ($\mathcal{C}_7$).

5.1.4 Applications

Langevin models are particularly suited to systems with an underlying physical model, as these relate to the deterministic function. Examples are coupled oscillators, analogue electrical circuits and potential wells. Langevin models are also suited to describing a stochastic process on a macroscopic scale, where this results from many microscopic subsystems that are self-organising [243, 244, 236]. Langevin models have gained recent popularity for their ability to describe data in a wide range of applications. Examples include meteorological systems [231], physiological systems [232, 223], financial markets [245], traffic flow [230] and measurement of rough surfaces [224].

In one example of physiological data analysis, Kuusela *et al.* [232] found that the Langevin model describes heart rate fluctuations over characteristic time scales of tens of minutes. Since this discovery the Langevin model for heart rate fluctuations has been used for data classification and proved successful at distinguishing between healthy and diseased patients [223]. In an example of spatial data analysis, Jafari *et al.* [224] demonstrated that the undulations on metallic surfaces within microscopic data are well described by a Langevin process. The authors demonstrated the simulation of realistic surfaces using the Euler-Mayarama scheme with a trained model.

5.2 Gaussian Process Models

Gaussian processes (GPs) are a general method for modelling the prior distribution and estimating the posterior distribution over discrete functions. Gaussian processes are naturally suited to tasks involving observations in state space. As a result the models are often seen as a regression tool [246]. Gaussian processes are also gaining popularity in the field of time series analysis. In chapter 7 we extend this idea for the case of radial time series, to create a global shape model ($\mathcal{C}_3$). In the context of a segmentation framework the 'regression' view also extends to incorporating observations from the image model ($\mathcal{C}_2$) and interactions ($\mathcal{C}_7$) in chapter 8.

The model treats a time series as a random vector of outputs $\mathbf{x} = \{x_0, \ldots, x_i, \ldots, x_{N-1}\}$ corresponding to inputs at discrete time points $\mathbf{t} = \{t_0, \ldots, t_i, \ldots, t_{N-1}\}$. The output at each t_i has an associated probability $\Pr(x_i|t_i)$, which we assume to follow a normal distribution. As a result the vector of outputs has a multivariate normal distribution

$$\mathbf{x} \sim \mathcal{N}_{\mathrm{N}}(\mu, \mathbf{\Sigma}(\mathbf{x}, \mathbf{x})) \tag{5.11}$$

where μ is the discretised mean function given by the vector of expectation values $E(x_i)$ and $\mathbf{\Sigma}(\mathbf{x}, \mathbf{x})$ is the $N \times N$ matrix of covariances between pairs of outputs $\{x_i, x_j\} \forall i, j \in \{0, \ldots, N-1\}$, written as

$$\mathbf{\Sigma}(\mathbf{x}, \mathbf{x}) = \begin{pmatrix} \varepsilon_{0,0}(x_0, x_0) & \varepsilon_{0,1}(x_0, x_1) & \cdots & \varepsilon_{0,N-1}(x_0, x_{N-1}) \\ \varepsilon_{1,0}(x_1, x_0) & \varepsilon_{1,1}(x_1, x_1) & \cdots & \varepsilon_{1,N-1}(x_1, x_{N-1}) \\ \vdots & \vdots & \ddots & \vdots \\ \varepsilon_{N-1,0}(x_{N-1}, x_0) & \varepsilon_{N-1,1}(x_{N-1}, x_1) & \cdots & \varepsilon_{N-1,N-1}(x_{N-1}, x_{N-1}) \end{pmatrix}. \tag{5.12}$$

The covariance between each pair of outputs is taken as a function of the corresponding inputs $\varepsilon(x_i, x_j) = f(t_i, t_j)$ where f is also known as a kernel function. The mean function μ and the covariance function $\varepsilon(x_i, x_j)$ completely define the prior model of a GP.

As in the Langevin case, we wish to 'score' unseen radial time series according to the probability that they belong to a shape model. In section 7.3.3 we will see that a probabilistic shape score can take advantage of the fact that GPs are based on multivariate normal distributions.

5.2.1 Parameter estimation

Williams and Rasmussen [247] view GP regression as a machine learning tool where the task is to fit a discrete function μ and a parametric kernel function $\varepsilon(x_i, x_j, \mathbf{a})$ to training data, where $\mathbf{a} =$

$\{a_0 \dots a_p, \dots a_{P-1}\}$ is a vector of p parameters. In the context of shape modelling, estimating model parameters from multiple radial time series amounts to learning their common dynamical properties.

A full analysis requires that the functions and parameters are estimated simultaneously, with an optimisation scheme such as by Bayesian model selection [246]. For our purposes we assume knowledge of the functional forms of μ and $\varepsilon(x_i, x_j, \mathbf{a})$, and consider the task of estimating parameters $\mathbf{a}$ used in the covariance matrix $\mathbf{\Sigma}(\mathbf{x}, \mathbf{a})$ that best describes an observed series. This is the approach taken by several authors [247, 248, 249]. Recall that $\varepsilon(x_i, x_j, \mathbf{a})$ is actually evaluated in terms of input sites t_i and t_j, which are known. This means that the only unknowns, when building the covariance matrix, are the parameters $\mathbf{a}$, so we write $\mathbf{\Sigma}(\mathbf{a})$ for convenience.

The most common method of parameter estimation follows the approach usually accredited to Mardia and Marshall [250], but presented earlier by Kitanidis [251]. The approach is to express the likelihood of observed data in terms of the parametric covariance function, and find the parameters that maximise this likelihood. The probability of an observed series $\mathbf{x} = \{x_0, \dots, x_i, \dots, x_{N-1}\}$ follows the multivariate normal distribution

$$\Pr(\mathbf{x}|\mathbf{a}) = \frac{1}{2\pi^{\frac{N}{2}}|\mathbf{\Sigma}(\mathbf{a})|^{\frac{N}{2}}} \exp[-\frac{1}{2}(\mathbf{x}-\mu)^T\mathbf{\Sigma}^{-1}(\mathbf{a})(\mathbf{x}-\mu)] \tag{5.13}$$

where vector μ is the known discrete mean function and $\mathbf{\Sigma}(\mathbf{a})$ is the covariance matrix as in equation 5.2. Taking the negative log of equation 5.13 yields the cost function

$$\begin{aligned} L &= -\log(\Pr(\mathbf{x}|\mathbf{a})) \\ &= \frac{N}{2}\log(2\pi) + \frac{1}{2}\log(|\mathbf{\Sigma}(\mathbf{a})|) + \frac{1}{2}\big((\mathbf{x}-\mu)^T\mathbf{\Sigma}^{-1}(\mathbf{a})(\mathbf{x}-\mu)\big) \end{aligned} \tag{5.14}$$

which is to be minimised to find the most likely parameters $\mathbf{a}$. The authors in [250] and [251] use iterative gradient descent methods of the general form

$$\mathbf{a}_{k+1} = \mathbf{a}_k - \delta\mathbf{R}_k\frac{\partial L}{\partial \mathbf{a}_k}, \tag{5.15}$$

where k is the iteration number, δ is a step length which, in the case of the Levenberg-Marquardt algorithm, is made adaptive (δ_k) to ensure a decrease in the cost function at each iteration, and the matrices $\mathbf{R}_k$ and $\frac{\partial L}{\partial \mathbf{a}_k}$ are discussed next.

The matrix $\mathbf{R}_k$ governs the type of gradient descent algorithm. If $\mathbf{R}_k$ is the unit matrix, then 5.15 is the method of 'steepest descent'. If $\mathbf{R}_k$ is (an approximation of) the inverse of the second derivative $\frac{\partial^2 L}{\partial \mathbf{a}_k^2}$, then 5.15 is the (quasi-)Newton method. More commonly, authors use the Gauss-Newton method whereby $\mathbf{R}_k$ is the inverse of the $P \times P$ Fisher information matrix $\mathbf{F}$ evaluated for the current parameters $\mathbf{a} = \mathbf{a}_k$, having elements

$$\mathbf{F}_{m,n} = \frac{1}{2}\,\mathrm{Tr}\left(\mathbf{\Sigma}^{-1}(\mathbf{a})\frac{\partial \mathbf{\Sigma}(\mathbf{a})}{\partial a_m}\mathbf{\Sigma}^{-1}(\mathbf{a})\frac{\partial \mathbf{\Sigma}}{\partial a_n}\right), \tag{5.16}$$

where $\frac{\partial \mathbf{\Sigma}(\mathbf{a})}{\partial a_m}$ is the matrix of element-wise partial derivatives of the covariance matrix with respect to the m^{th} parameter.

The matrix $\frac{\partial L}{\partial \mathbf{a}_k}$ is the $P \times 1$ vector of partial derivatives of the likelihood with respect to the current parameters $\mathbf{a} = \mathbf{a}_k$. It follows from the product rule and properties of the trace of a matrix, that

$$\begin{aligned}\left(\frac{\partial L}{\partial \mathbf{a}}\right)_m &= \left(\frac{\partial L}{\partial a_m}\right) \\ &= \frac{\partial L}{\partial \boldsymbol{\Sigma}(\mathbf{a})}\frac{\partial \boldsymbol{\Sigma}(\mathbf{a})}{\partial a_m} \\ &= \frac{1}{2}\,\mathrm{Tr}\left(\boldsymbol{\Sigma}^{-1}(\mathbf{a})\frac{\partial \boldsymbol{\Sigma}(\mathbf{a})}{\partial a_m}\right) - \frac{1}{2}\left(\mathbf{x}^T\boldsymbol{\Sigma}^{-1}(\mathbf{a})\frac{\partial \boldsymbol{\Sigma}(\mathbf{a})}{\partial a_m}\boldsymbol{\Sigma}^{-1}(\mathbf{a})\mathbf{x}\right)\end{aligned} \tag{5.17}$$

Mardia and Marshall's gradient descent method is prone to converging on local minima [252]. Also, the calculation of $\mathbf{R}_k$ and the matrices of partial derivatives limits the suitability of the gradient descent method in the context of learning from radial time series for shape modelling. This is because equation 5.17 assumes that all training data belongs to a single series. While data from multiple training shapes could be concatenated, this would lead to impractically large matrices.

Markov Chain Monte Carlo (MCMC) methods offer an alternative approach as used in [247]. MCMC is a general machine learning technique, already discussed in chapter 2 for optimising deformable contours. In the context of parameter estimation the algorithm uses Monte Carlo sampling to draw parameters from the stable distribution that maximises the log likelihood $\log(\Pr(\mathbf{x}|\mathbf{a}))$, i.e. the negative of equation 5.14. Neal [248] states that MCMC methods are the only feasible approach to parameter estimation, especially for larger numbers of parameters.

The MCMC method is particularly suited to our purposes, of learning common dynamical properties from multiple shapes (radial time series) in a class. This type of machine learning is referred to as 'multitasking' [253, 246]. In section 7.3.2 we exploit the joint probability $\Pr(\mathbf{x}^{m=0,\ldots,M}|\mathbf{a})$ of observing the set of M series denoted $\mathbf{x}^{m=0,\ldots,M-1}$. The joint log probability L', i.e. the thing to maximize, becomes a summation

$$\begin{aligned}\Pr(\mathbf{x}^{m=0,\ldots,M-1}|\mathbf{a}) &= \prod_{m=0}^{M-1}\Pr(\mathbf{x}^m|\mathbf{a}) \\ &= \prod_{m=0}^{M-1}\frac{1}{2\pi^{\frac{N}{2}}|\boldsymbol{\Sigma}(\mathbf{a})|^{\frac{N}{2}}}\exp[-\frac{1}{2}(\mathbf{x}^m-\mu)^T\boldsymbol{\Sigma}^{-1}(\mathbf{a})(\mathbf{x}^m-\mu)] \\ L' &= \log\Pr(\mathbf{x}^{m=0,\ldots,M-1}|\mathbf{a}) \\ &= -\frac{MN}{2}\log(2\pi) - \frac{M}{2}\log(|\boldsymbol{\Sigma}(\mathbf{a})|) - \frac{1}{2}\sum_{m=0}^{M-1}\big((\mathbf{x}^m-\mu)^T\boldsymbol{\Sigma}^{-1}(\mathbf{a})(\mathbf{x}^m-\mu)\big),\end{aligned} \tag{5.18}$$

and an MCMC algorithm is set up to draw samples from the stable distribution $\pi(\mathbf{a})$ having expectation value equal to the maximum *a-posteriori* probability (MAP) estimate of $\mathbf{a}$.

5.2.2 Generative model

This section defines standard procedures for generating GPs [246]. Generating a series from the model is equivalent to drawing a random vector of outputs $\mathbf{x} = \{x_0, \ldots, x_i, \ldots, x_{N-1}\}$ from the prior distribution. After developing GPs for shape modelling the generative models form the basis of probabilistic

segmentation frameworks in section 8.5.

If $\mathbf{z}$ is a N-dimensional vector of independent variables, each drawn from $\mathcal{N}(0,1)$ then we can write

$$\mathbf{z} \sim \mathcal{N}_N(\mathbf{0}, \mathbf{I}) \tag{5.19}$$

where $\mathbf{0}$ is a vector of zeros and $\mathbf{I}$ is the $N \times N$ identity matrix. If the covariance matrix $\mathbf{\Sigma}(\mathbf{x}, \mathbf{x})$ is positive semidefinite, it follows that

$$\mathbf{x} = \mu + \mathbf{A}\mathbf{z} \sim \mu + \mathcal{N}_N(\mu, \mathbf{A}\mathbf{A}^\mathbf{T}), \tag{5.20}$$

where $\mathbf{A}$ is the Cholesky decomposition of $\mathbf{\Sigma}(\mathbf{x}, \mathbf{x})$. Calculating $\mathbf{x}$ from equation 5.20 requires the following steps:

Step 1. Form a vector of N discrete inputs $\mathbf{t} = \{t_0, \ldots, t_i, \ldots, t_{N-1}\}$.

Step 2. Construct the covariance matrix $\mathbf{\Sigma}(\mathbf{x}, \mathbf{x})$ by evaluating elements in equation 5.2 using the kernel function $\varepsilon(x_i, x_j) = f(t_i, t_j)$.

Step 3. Take the Cholesky decomposition of $\mathbf{\Sigma}$ to give $\mathbf{A}$.

Step 4. Construct $\mathbf{z}$ by generating N random variables z_i from the normal distribution $\mathcal{N}(0,1)$.

Step 5. Calculate $\mathbf{x} = \mu + \mathbf{A}\mathbf{z}$.

This procedure generates a whole series simultaneously, according to the prior dynamical model defined by the mean function and covariance kernel. The next section describes extensions to draw samples from the posterior model in the light of observed data.

5.2.3 Incorporating observations

As proposed above, in the context of radial time series modelling in segmentation, the image model ($\mathcal{C}_2$) and interactions ($\mathcal{C}_7$) can play the role of observations. It is desirable for a generated series to combine these observations with the dynamics defined by the mean and covariance functions. In the GP model this type of constraint amounts to conditioning the prior over function space. Generating series in turn amounts to drawing samples from the posterior. Here we describe the procedures for conditioning the prior in the light of observations with and without associated noise. The noisy and noise free observations are represented by image models in section 8.5.1 and interactions in section 8.5.2.1.

5.2.3.1 Noise free observations

GP regression is based on conditioning the prior model on observations x_i at corresponding inputs t_i. We re-write the vector of inputs above as $\mathbf{t}^* = \{t_0^*, \ldots, t_i^*, \ldots, t_{N^*-1}^*\}$, where $*$ denotes that the corresponding outputs are unknown. These outputs are denoted $\mathbf{x}^* = \{x_0^*, \ldots, x_i^*, \ldots, x_{N^*-1}^*\}$ and could be predicted by the GP using the generative model above. For N observations we have the vector of known outputs $\mathbf{x} = \{x_0, \ldots, x_i, \ldots, x_{N-1}\}$ at inputs $\mathbf{t} = \{t_0, \ldots, t_i, \ldots, t_{N-1}\}$. Consider a new vector $\mathbf{v} = \{\mathbf{t}, \mathbf{t}^*\}^T$, which is constructed by concatenating input vectors for the N observations and N^*

unknown outputs. This vector has length $N' = N + N^*$ and covariance matrix

$$\begin{aligned}\mathbf{\Sigma}(\mathbf{v},\mathbf{v}) &= \begin{pmatrix} \varepsilon_{0,0}(x_0,x_0) & \dots & \varepsilon_{0,N}(x_0,x^*_{N'-1}) \\ \vdots & \ddots & \vdots \\ \varepsilon_{N,0}(x^*_{N'-1},x_0) & \dots & \varepsilon_{N,N}(x^*_{N'-1},x^*_{N'-1}) \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{\Sigma}_{\mathbf{x},\mathbf{x}} & \mathbf{\Sigma}_{\mathbf{x},\mathbf{x}^*} \\ \mathbf{\Sigma}_{\mathbf{x}^*,\mathbf{x}} & \mathbf{\Sigma}_{\mathbf{x}^*,\mathbf{x}^*} \end{pmatrix},\end{aligned} \tag{5.21}$$

where $\mathbf{\Sigma}_{\mathbf{x},\mathbf{x}}$ denotes a $N \times N$ sub matrix, $\mathbf{\Sigma}_{\mathbf{x},\mathbf{x}^*}$ is a $N \times N^*$ sub matrix and so on. These sub-matrices are used to draw samples from the posterior as follows.

The posterior has a multivariate normal distribution of dimension N^*, written as

$$\Pr(\mathbf{x}^*|\mathbf{t}^*,\mathbf{t},\mathbf{x}) = \mathcal{N}_{N^*}(\mu_{\text{post}},\mathbf{\Sigma}_{\text{post}}), \tag{5.22}$$

where μ_{post} is the $N^* \times 1$ posterior mean function and $\mathbf{\Sigma}_{\text{post}}$ is the $N^* \times N^*$ posterior covariance matrix.

Calculation of the posterior mean function μ_{post} involves accounting for known observations $\mathbf{x}$ by

$$\mu_{\text{post}} = \mu + \mathbf{\Sigma}_{\mathbf{x}^*,\mathbf{x}}\mathbf{\Sigma}^{-1}_{\mathbf{x},\mathbf{x}}(\mathbf{x} - \mu_x). \tag{5.23}$$

where μ_x is the prior mean function evaluated at the same input sites as the observations $\mathbf{x}$.

Calculation of the posterior covariance matrix $\mathbf{\Sigma}_{\text{post}}$ involves all pairs of inputs with both known and unknown outputs, by

$$\mathbf{\Sigma}_{\text{post}} = \mathbf{\Sigma}_{\mathbf{x}^*,\mathbf{x}^*} - \mathbf{\Sigma}_{\mathbf{x}^*,\mathbf{x}}\mathbf{\Sigma}^{-1}_{\mathbf{x},\mathbf{x}}\mathbf{\Sigma}(\mathbf{x},\mathbf{x}^*). \tag{5.24}$$

Finally, equation 5.20 is replaced by

$$\mathbf{x}^* = \mu_{\text{post}} + \mathbf{A}_{\text{post}}\mathbf{z} \sim \mu_{\text{post}} + \mathcal{N}_{N^*}(\mu_{\text{post}},\mathbf{A}_{\text{post}}\mathbf{A}^T_{\text{post}}), \tag{5.25}$$

where $\mathbf{z} \sim \mathcal{N}(\mathbf{0},\mathbf{I}^*)$ with $\mathbf{I}^*$ being a $(N^* \times N^*)$ identity matrix, and $\mathbf{A}_{\text{post}}$ is the Cholesky decomposition of $\mathbf{\Sigma}_{\text{post}}$. Samples drawn from the posterior in equation 5.25 are conditioned to pass through the observed points. When working with radial time series models for segmentation in chapter 8, we will adapt the constraint of noise-free observations in order to condition shape models in response to user interactions ($\mathcal{C}_7$).

5.2.3.2 Noisy observations

In many practical cases, an observation at a given time t_i is represented by a mean $\hat{x}_i$ and variance σ_i^2. GP models have also been developed to condition the prior on these 'noisy' observations. The method stores observation means and variances in a vector $\hat{\mathbf{x}}$ and a matrix $\sigma^2\mathbf{I}$ respectively, where $\mathbf{I}$ is a $N \times N$ identity matrix and $\sigma^2 = \{\sigma_0,\dots,\sigma_{N-1}\}$ are the independent variances of each of N noisy observations. Because of the independence assumption we can add the variance matrix $\sigma^2\mathbf{I}$ to the covariance matrix of the joint distribution given in equation 5.21 to give

$$\mathbf{\Sigma}(\mathbf{v},\mathbf{v}) = \begin{pmatrix} \mathbf{\Sigma}_{\mathbf{x},\mathbf{x}} + \sigma^2\mathbf{I} & \mathbf{\Sigma}_{\mathbf{x},\mathbf{x}^*} \\ \mathbf{\Sigma}_{\mathbf{x}^*,\mathbf{x}} & \mathbf{\Sigma}_{\mathbf{x}^*,\mathbf{x}^*,} \end{pmatrix}, \tag{5.26}$$

where, as in section 5.2.3.1, the covariance matrix $\mathbf{\Sigma}(\mathbf{v}, \mathbf{v})$ divided into four sub-matrices used in the predictive equation

$$\Pr(\mathbf{x}^*|\mathbf{t}^*, \mathbf{t}, \hat{\mathbf{x}}, \sigma^{\mathbf{2}}) = \mathcal{N}_{N^*}(\mu_{\text{post}}, \mathbf{\Sigma}_{\text{post}}), \tag{5.27}$$

where

$$\mu_{\text{post}} = \mu + \mathbf{\Sigma}_{\mathbf{x}^*,\mathbf{x}}[\mathbf{\Sigma}_{\mathbf{x},\mathbf{x}} + \sigma^{\mathbf{2}}\mathbf{I}]^{-1}(\hat{\mathbf{x}} - \mu). \tag{5.28}$$

and

$$\mathbf{\Sigma}_{\text{post}} = \mathbf{\Sigma}_{\mathbf{x}^*,\mathbf{x}^*} - \mathbf{\Sigma}_{\mathbf{x}^*,\mathbf{x}}[\mathbf{\Sigma}_{\mathbf{x},\mathbf{x}} + \sigma^{\mathbf{2}}\mathbf{I}]^{-1}\mathbf{\Sigma}_{\mathbf{x},\mathbf{x}^*}. \tag{5.29}$$

Samples drawn from the posterior in equation 5.27 are attracted the the observation means, more so for those having smaller variance. In the context of radial time series models in segmentation, chapter 8 adapts these ideas to derive noisy observations from boundary-based image models ($\mathcal{C}_2$).

5.2.4 Applications

GP models are mainly used for regression and classification tasks. The role of GPs as a regression tool, fitting functions to sparse and noisy data, makes this modelling technique suited to the contouring problem where the observation series (boundary measure) is noisy and may contain gaps. The role of GPs as a classification tool extends to the use of a GP prior in segmentation by regularisation.

The use of GP models for time series modelling is well established. Recent examples in computer vision use the dynamical priors to improve object tracking [254, 255]. In these cases the approach is viewed as a nonlinear extension to methods using linear autoregressive models, such as that in [256]. Wang *et al.* [254] introduced a GP framework to model higher dimensional time series for tracking human motion over video sequences. Urtasun *et al.* [255] later showed that this framework can learn complex dynamics with modest amounts of training data, whilst handling occlusion in the observation series and low image quality.

5.3 Discussion and Conclusions

This chapter reviewed two time series models, chosen as extensions to those used for shape modelling and segmentation in the vision literature. Langevin models share the Markov property with the 1D-CMRF used in shape classification and segmentation. We view a 1-dimensional GP as a nonlinear extension of the autoregressive time series model [254, 255].

The wider field of time series analysis involves other types of dynamical model, including chaotic and purely statistical models, which are also used biomedical data analysis. Our choice of model is governed by two main questions. The first is what we can assume about the data. For example, we make no *a priori* assumptions as to whether boundary fluctuations for tumorous regions of interest are Markovian. As such we have chosen two models, Langevin and GP, which respectively do and do not assume this property. We make no *a priori* assumptions as to whether radial time series exhibit chaotic behaviour. We learn from the example of heart rate fluctuations, that the same physiological time series can be described by both chaotic [257, 258] and stochastic [232] models. The second question is what we want the model to do. As we have already justified, we desire a model that provides hypotheses, in

other words generate shapes with model dynamics for use in segmentation algorithms. Purely statistical models such as those based on fractal analysis [259] or symbolic complexity [260] or other entropy measures [261, 262, 263] do not readily offer this functionality.

In discussing the relevance of Langevin and GP models for shape modelling we note the key similarities and differences between the two approaches. The models are similar in that they both offer discriminative and generative methods that, if combined with the radial time series boundary representation, lend themselves to statistical shape modelling. Both models represent a series by a deterministic function and an independent stochastic sequence, and so could be used in frameworks with a stochastic deformation mechanism and optimisation scheme. In the case of Langevin models, the delay parameter Δt is the characteristic timescale at which fluctuations can be considered Markovian. This is analogous to the width parameter of a GP kernel. These parameters in turn play the role of the number m of 'lag terms' in an autoregressive (AR) model. Finally, Langevin and GP models are both nonlinear and as such, following the conclusions made in [165], expected to model complex shapes having high within-class variability. We discuss differences between the models in the next subsection.

5.3.1 Differences between Langevin and GP models

The Langevin approach models Markovian dynamics, based on a stationary function of the local clique $\{x_i, x_{i-n}\}$. As a result, the machine learning methods in a Langevin model can make use of incomplete training contours, or the method adapted for open contour models. Conversely, GPs are based on a continuous correlation function over all x and must be trained on complete series.

As a result of their Markovian nature, Langevin models do not readily represent cyclic series. GP models on the other hand can model cyclic series by the choice of a periodic kernel function. This is relevant to radial time series that exists over a periodic range of 2π radians.

In the case of Langevin models, observations are not easily included by current methods, as so-called 'data assimilation' is the subject of ongoing research. Conversely, GP models naturally use observations to condition the prior model, as this constitutes the regression task for which GPs were primarily developed.

Another difference lies in the generative methods for simulating a series from each case. The SDE method of Langevin models generates a series in successive steps. In the context of deformable contours, this is analogous to the deformation mechanisms specific to boundary tracking. The GP generative model on the other hand draws the whole of a series from the prior model in *one-shot*.

Finally, unlike Gaussian processes, Langevin-type series assume an underlying physical model. Examples in the literature suggest that these assumptions hold in many real world applications, including physiological processes and one example of a series of points in the spatial domain. Kleinhans *et al.* highlights the suitability of Langevin methods for modelling complex systems of physics, chemistry and biology [236], where macroscopic series are the result of complex interactions between microscopic subsystems. This applies to the complex physiological mechanisms that give rise to regions such as tumours and lesions in medical images, as backed up by simulations based on physical models of constrained diffusion in biomechanical literature (eg. [264]).

This is also in line with other ways in which physical models have been used to constrain segmentation and the related field of image registration. In the case of segmentation, biomechanical knowledge allows tailored definitions of internal energy. In [265] for example, the authors segment cellular regions in single plane illumination microscopy (SPIM) images using a smoothness constraint based on analytical expressions for the 'lipid bilayer bending energy' in cell membranes. In the case of registration, biomechanical models have been used to constrain the deformation field that aligns two images, by incorporating knowledge of the physical processes underlying regional differences in the two images [75, 266].

In conclusion, the field of time series analysis outside the segmentation literature is vast and has seen rapid development since the use of CAR and 1D-CMRF for shape classification. We have chosen to review GP and Langevin models, both of which are nonlinear, flexible and capable of modelling natural phenomena in an intuitive way. Both models also introduce robust schemes for machine learning, series simulation/generation and, in the GP case, the incorporation of an observation model. We are therefore motivated to develop these approaches for statistical shape modelling and supervised segmentation. With a radial time series representation, the Langevin and GP approaches can be seen as extensions to the linear CAR and 1D-CMRF models respectively. The extensions also introduce nonlinearity and higher-level information known to benefit a segmentation framework and reduce the amount of on-line information needed (*Requirement 2*). We identify the following requirements to enable novel shape priors based on Langevin and GP models:

- Developing 'shape scoring' methods for use in (classification and) an objective function.
- Adapting GP data assimilation procedures to incorporate observations from an image model and interactions.
- Devising Langevin data assimilation procedures to incorporate observations from an image model and interactions.
- Choosing deterministic functions and machine learning procedures appropriate for the training data available.
- Devising methods of generating periodic series in the Langevin case.
- Building generalised DCM frameworks driven by generative SSMs, capable of incorporating these or other models.

Chapter 6

Tracking Ambiguous Boundaries

This chapter seeks to improve supervised segmentation, by introducing prior knowledge driven by efficient run-time interactions ($\mathcal{C}_7$) and novel image models ($\mathcal{C}_2$). Chapter 2 concluded that the most efficient modes of interaction work closely with other components of the segmentation framework, and proposed that boundary tracking approaches offer the optimal balance between a user's control and involvement. Chapter 4 concluded that the SVM 'kernel trick' leads to image classification methods that generalise across applications and maximise the information gained from monospectral data. This chapter develops these two key ideas by combining interactive boundary trackers and texture classification in novel segmentation frameworks.

Section 6.1 develops algorithms for boundary tracking and introduces an interactive framework. Experiments in section 6.1.5 evaluate the useability of the interactive framework in user trials. Section 6.2 investigates the use of SVMs for texture classification and boundary extraction in low contrast images. Experiments in section 6.2.5 test the efficacy of the SVMs for the boundary ambiguity problem in synthetic and medical images. Section 6.3 shows how the boundary trackers can be driven by the output of SVM classifiers to reduce problems associated with boundary ambiguity. Experiments in section 6.3.2 and 6.3.3 evaluate the benefits of the SVM in supervised segmentation of synthetic and MS lesion images respectively.

6.1 Interactive Boundary Tracking

This section develops an interactive boundary tracking tool based on the jetstream algorithm of Perez *et al* [43], with novel modes of interaction ($\mathcal{C}_7$) designed to increase the amount of user control (*Requirement 1*). Sections 6.1.1 and 6.1.2 present adaptions made to the original jetstream algorithm and other interactive methods built into the framework. Section 6.1.3 introduces two new algorithms, constrained to terminate the open contour at a fixed point provided interactively. These algorithms are intended as methods for completing a closed contour during segmentation and we choose from the two on the basis of qualitative evaluations. Sections 6.1.4 and 6.1.5 describe how we evaluate the interactive software and present experiments that test the novel interactions and loop closing.

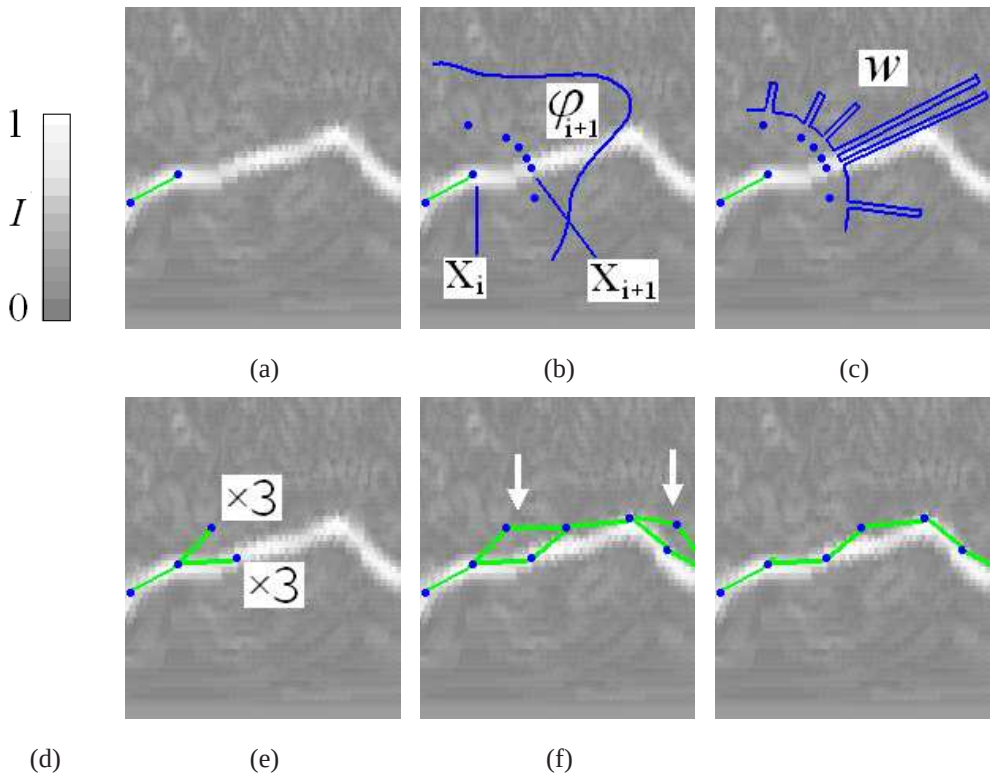


Figure 6.1: Schematic diagram of boundary tracking by particle filtering. Grey-levels represent any image model I, such as gradient magnitude, or SVM decision value, rescaled to the range $\{0, \ldots, 1\}$. (a) The current 'step' comprising boundary points (blue) interpolated with a straight line (green). (b) The *prediction* stage, making $M = 6$ proposals for the point $\mathbf{x}_{i+1}$ by drawing from the prior distribution of angles φ. (c) The *weighting* stage, forming a discrete estimation of the posterior by weights w. (d) The *importance sampling* stage causes some predictions to be duplicated and some to be discarded. In this example two predictions have survived and are each duplicated three times. (e) Several steps create several tracking paths that share many points. Arrows show where particles have different $\mathbf{x}_i$. (f) The path with highest overall weight gives the desired estimate of the contour section.

6.1.1 A particle filtering algorithm

Recall from section 2.2.1, the 'jetstream' algorithm works with a set of M contours $(\mathbf{x}^m_{0,\ldots,i})_{m=0,\ldots,M-1}$ where $\mathbf{x}_i = \{x, y\}$ is a position vector in the image frame. The algorithm tracks a boundary section $\mathbf{x}_{0,\ldots,i}$ by making iterative estimates of the posterior densities

$$p_{i+1}(\mathbf{x}_{0,\ldots,i+1}|\mathcal{D}) \propto p_i(\mathbf{x}_{0,\ldots,i}|\mathcal{D}) \times q(\mathbf{x}_{i+1}|\mathbf{x}_{i-1,\ldots,i}) \times l(\mathcal{D}(\mathbf{x}_{i+1})), \tag{6.1}$$

where $\mathcal{D}$ denotes information in the image and q and l are the smoothness prior and data likelihood respectively, given as functions of the angle made by a given 'step' from $\mathbf{x}_i$ to $\mathbf{x}_{i+1}$ and the local image properties. Section 2.2.1 gave a full description of these functions and the particle filtering algorithm, comprising stages of *prediction*, *weighting* and *importance sampling*. Figure 6.1 illustrates these stages.

We make three technical adaptations to the jetstream algorithm based on intuition. First, we replace the normal prior angular distribution, $q = \mathcal{N}(\varphi, 0, \sigma_\varphi)$, with the Von-Mises distribution[267]

$q = \mathcal{V}(\varphi, 0, \kappa_\varphi) \propto \exp(\kappa_\varphi cos(\varphi - 0))$, which is periodic with period 2π and spread controlled by κ_φ. This distribution is better suited to model the prior over a periodic variable like φ. Second, we constrain boundary tracking to avoid self-intersection. This is done by keeping a map of the contour sections accepted from previous runs. For any proposal step that lands on or traverses the pixels in this map, we repeat the prediction stage of drawing samples from $\mathcal{V}(\varphi, 0, \kappa_\varphi)$, with increasing variance σ_φ, until M proposals do not intersect the contour. The algorithm then proceeds by weighting these proposals and resampling from them as usual. Third, we define a jetstream as the particle set that, after importance sampling, has the maximum total weight, i.e. the highest value of $\sum_{i=0}^{N-1} w$. The authors in [43] argue that the mean path, rather than the path of maximum weight, is a 'more stable' solution. However, the averaging process can smooth out desired boundary detail and could create false paths where the posterior is multi-modal such as at forks in the true boundary. The arrows in figure 6.1 (e) show parts of a boundary where taking the mean over particles would cause unnecessary deviation from the true boundary, caused by nearby clutter giving high weights.

6.1.2 An interactive framework

We extend the initialisation and run-time interactions suggested in [43] to make more efficient use of the information provided. Figure 6.2 demonstrates the framework, used for the application of MS lesion contouring. Rather than a single anchor, we initialise with a small straight-line (2 or more pixels). This provides both $\mathbf{x}_0$ and the initial direction φ_0 in figure 6.2 (b) without placing much additional demand on the user. This mode of initialisation can also increase speed and user-control if the jetstream can be initialised at a long, straight section of a boundary. Upon initialisation, the software displays the contour section resulting from a single run $\mathbf{x}_{0,\ldots,N-1}$, with points x_i interpolated using Bresenham's line algorithm [268]. Contouring continues by making efficient use of anchors placed around the boundary. In [43], as with live wires in [90], the anchor simply fixes the contour model at the last accepted point on the boundary. The methods described next, (section 6.1.2.1) provide a fully *user steered* framework with increased control. This type of steering is shown to work in a real-world segmentation in figure 6.2, and replaces extra measures used in [43] for handling sharp corners in a boundary. The final run of the jetstream is constrained to terminate at the starting point as in figure 6.2(d). Loop closing methods are revisited in section 6.1.3.

6.1.2.1 User guided contours

Figure 6.3 illustrates the modes of interaction that a user can use to guide a jetstream contour along a boundary. At any time the contour is made up of accepted sections from successive runs. Between runs, the user selects an anchor point, from where the next run begins. An anchor placed at the *tip* of the jetstream causes the tracking to continue as normal (figure 6.3, top row). If an anchor is placed *beyond* the tip, the Bresenham algorithm interpolates a straight line from the tip to the anchor and the next run begins from the anchor, with initial direction given by the interpolation (figure 6.3, second row).

If the user wishes to discard some of a jetstream, i.e. to correct for a divergence from the true boundary, an anchor is placed at the last accepted pixel *along* the contour (figure 6.3, third row). This anchor becomes part of the contour model and all points from there on-wards are removed. A new run

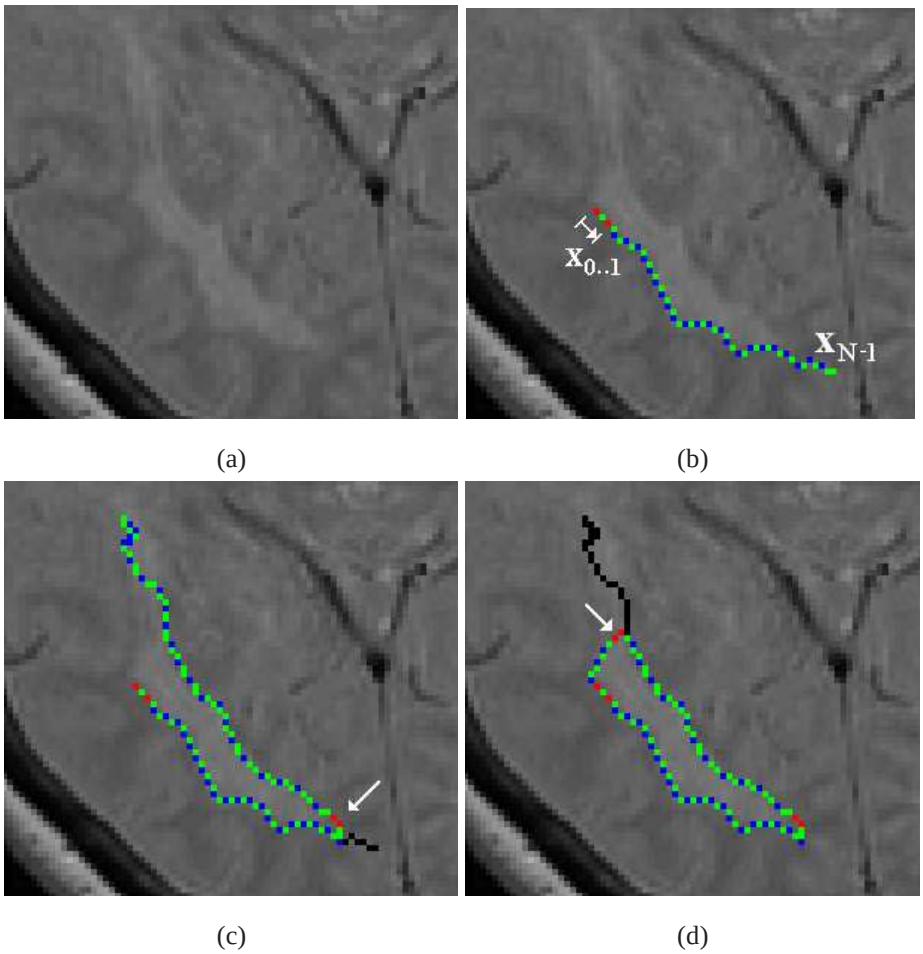
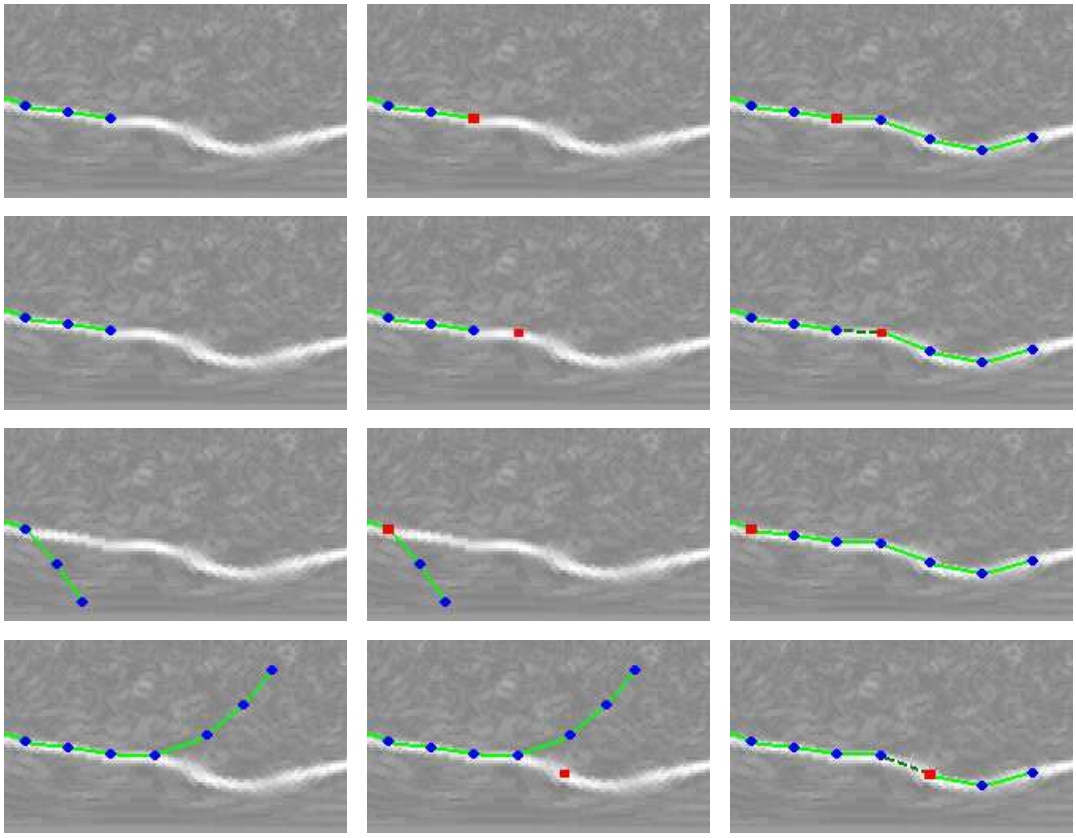


Figure 6.2: Annotated screenshots from a jetstream during segmentation. (a) A white matter lesion in a mid-axial MR image of the brain. (b) The first jetstream run comprising a straight user-defined section from $\mathbf{x}_0$ to $\mathbf{x}_1$ and N subsequent steps. Pixels selected by the user are shown in red, points $\mathbf{x}$ that make up the parametric contour in blue and their interpolation in green. (c) Second user interaction to steer the contour followed by N steps. (d) Final interaction that both steers the contour and enforces loop closure. Each arrow in (c) and (d) points to a pair of anchor points. Contour sections discarded by each interaction are shown as black lines in (c) and (d).

begins from the anchor, with initial direction given by the preceding step of the contour.

If an anchor is placed anywhere along, but slightly *to one side* of the current contour (figure 6.3, bottom row), subsequent points are removed as before but the last accepted point on the contour is not immediately obvious. To identify the last accepted point $\mathbf{x}_{\text{last}}$ we step through the points $\mathbf{x}$ and find the point that is closest to, but not beyond the anchor. We then interpolate from here to the anchor, which becomes the point $\mathbf{x}_{\text{last}+1}$ and the next run continues from there. The subsequent run has an initial direction governed by the interpolation from $\mathbf{x}_{\text{last}}$ to the anchor. This in turn is ultimately governed by the user upon placing the anchor. When a user is familiar with the tool, and aided by the visualisation of points $\mathbf{x}$ in their own colour (blue), this method of steering a contour between runs provides the main form of run-time supervision.



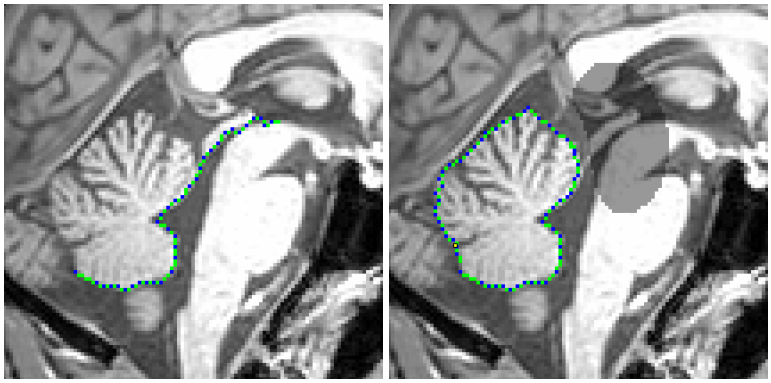
(a) (b) (c)

Figure 6.3: Schematic diagram illustrating four different roles of interactive 'anchoring' in a user-steered framework. Contour points $\mathbf{x}_i$ are shown in blue and their straight-line interpolation in light green. Dashed, dark green lines show where an interpolation joins separate runs rather than steps within a run. Columns show (a) the 'current' jetstream, (b) an anchor (red square) placed by the user and (c) subsequent steps of the jetstream in response to the anchor placement. *Top row*: the anchor is placed at the tip of the current jetstream. *Second row*: the anchor is placed beyond the tip. *Third row*: the anchor is placed at the last accepted point, along the contour that deviates from the desired path. *Bottom row*: the anchor is placed to one side of the contour, around the point of divergence.

6.1.2.2 Extra interactions

The software includes a panel of 'slidebars' for changing parameters of the contour model during run-time. The user can adjust the smoothness (κ_θ), detail (step length) and length (N) of subsequent runs using slide-bars. We also provide two extra methods of online supervision. The first is based on the 'damming' method in the original jetstream paper [43]. The authors allow false paths, such as at a fork in the boundary, to be blocked by drawing a small line on the image. We allow the user to draw a whole area, that the jetstream will subsequently avoid. We call these areas 'no-go' areas. By choosing an area rather than a line, nearby clutter (erroneous edges or false positive classifications) can be eliminated in the same interaction with little extra effort. A map of current nogo-areas is recorded in the same binary map as that used to avoid self-intersection. Figure 6.4 shows an example of the no-go area in use, during

a delineation of the cerebellum in a sagittal MRI slice of the brain. The user-defined area is dimmed in the image so that the user remembers where it is (and can reject it if need be). The false path taken by the jetstream in (a), without the no-go area, is avoided when the interaction is repeated with the no-go area (b).



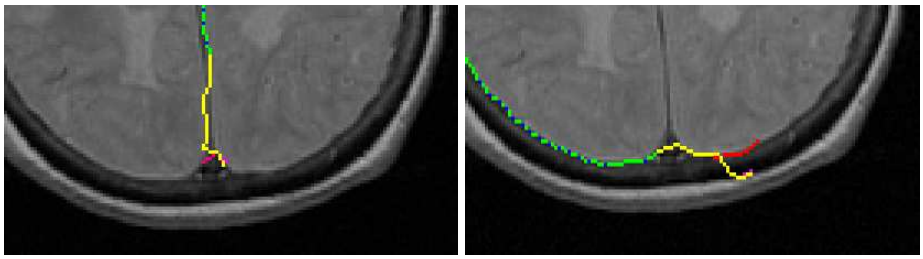
(a) (b)

Figure 6.4: Segmenting the cerebellum in a sagittal MR image of the author's brain. (a) A jetstream run is first performed without a 'no-go' area. (b) Repeating with a 'no-go' area, indicated by darkened pixels, excludes nearby clutter and avoids a false path.

We also introduce a novel interactive procedure that exploits the probabilistic nature of the underlying algorithm. At any one time the jetstream algorithm retains a set of M particles $\mathbf{x}_{i=0,\ldots,N-1}^{m=1,\ldots,M-1}$, of which only one gives the contour. The chosen path is that with the highest cumulative weighting along the N steps, but one of the *unseen* paths might better represent the desired boundary. This is likely when a boundary is close to false edges, causing the posterior distribution of proposal steps to be bimodal. We allow the user to manually select particles after each jetstream run, from the set $\mathbf{x}_{i=0,\ldots,N-1}^{m=1,\ldots,M-1}$. To do this we must display some of the unseen particles. In practise the particles are very similar for most of the run, but might differ for the last few steps. The method, invoked by the user, displays a small number of particles chosen for their distinction from the current contour. The particles are shown in different colours and the user simply selects one with the mouse. Figure 6.5 shows two examples of this procedure in use. In (a) the jetstream reaches a fork in the boundary of a large region (brain hemisphere) and the displayed particles follow left- and right-hand paths. In (b), some particles are attracted to a strong, false edge (skull). The red particle, however, successfully tracks the brain boundary.

6.1.3 Loop closing

Section 2.2 revealed that one of the problems facing boundary tracking algorithms is the lack of constraint to terminate at a fixed point. Such a constraint would allow easy closing of an open contour in the framework above, where the termination point is given by the first contour point $\mathbf{x}_0$. This method presents two methods and chooses one based on empirical investigations.



(a) (b)

Figure 6.5: Segmenting the brain making use of manual particle selection to correct for ambiguity in (a) path direction at a fork in the boundary, and (b) the true edge when a nearby erroneous edge is strong.

6.1.3.1 A method from tractography

Tractography aims to track white matter fibers in diffusion tensor imaging (DTI). Each voxel in DTI data gives information regarding the likelihood that a voxel is part of a fiber and the local direction of the fiber at that point. Knowledge of brain anatomy and function mean that we know approximately where a tract should originate and terminate. This knowledge can be built into a tractography algorithm to constrain fiber tracking. We propose that similar methods can constrain boundary tracking algorithms to terminate at a pre-defined point, hence close a contour.

Friman and Westin [269] use a tractography algorithm similar to particle filtering. The origin A of a fiber can be marked manually or automated using anatomical knowledge. During the process of growing a fiber tract, the dynamics of the model are not affected by the position of the target B. However, if there is a true path from A to B then tracts are likely to reach the target. The algorithm relies on this fact and repeats the tracking, from the same origin, until many tracts have terminated at B by chance. Those particles that do not reach the target within a sensible length are discarded. Finally a single path (or a population with associated probability) is extracted by importance sampling from the remaining set. This Monte Carlo approach is intuitive but computationally demanding, depending on the consistency of the path from A to B. Recently, Jbabdi *et al.* [270] take a similar approach wherein the likelihood of a growing tract terminating at a given point is driven by a map of effective connectivity between all points, measured separately by functional MRI. The constraints in both [269] and [270] can be referred to as 'soft' constraints, as the start and end points can be anywhere within small regions rather than asserting unique voxels A and B. This is where the tractography analogy to open contour segmentation breaks down.

We introduce an adaptation to the jetstream algorithm that causes the final run to terminate at the start of the contour to close the loop. The software assumes that the user anticipates when a run can close the loop and invokes the loop closing algorithm by pressing a key on the keyboard (similar to [42]). The method borrows from the tractography literature above. The underlying assumption is that a run is likely to reach the target point by chance. The loop closing run no longer has a fixed number of steps, but proceeds until enough particles are within one step-length of the first point $\mathbf{x}_0$. When this is true, it is likely that nearly all of the proposal steps $\mathbf{x}_{0,\ldots,i+1}$ arrive near the target pixel at the same time, as all

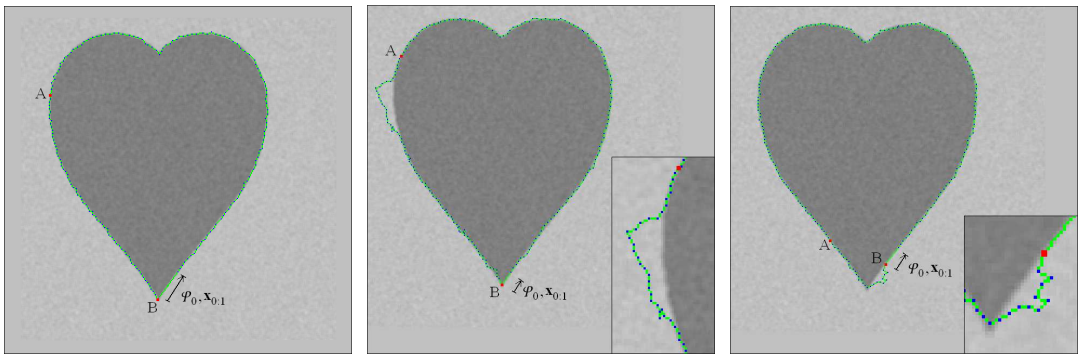
particles $\mathbf{x}_{0,\ldots,i}$ share similar histories $\mathbf{x}_{0,\ldots,i-1}$. The algorithm then uses importance sampling to select from these particles only.

With no fixed number of steps N, it is possible that particles diverge from the desired boundary, visiting clutter in the image before hitting the target point by chance. To avoid this we direct the particles toward $\mathbf{x}_0$ by the addition of an extra prior term in the weighting stage. The new weights w_{att} incorporate an *attraction* force by the definition

$$w_{\text{att}} = \frac{q \times l}{(i+1)} + (i+1)\exp\left(-\frac{D}{2}\right) \qquad i \in \{0, 1, \ldots, i_{\text{hit}}\}, \tag{6.2}$$

where D is the straight line distance from $\mathbf{x}_i$ to $\mathbf{x}_0$. The attraction force is given a relative weighting that increases with the step index i to help 'pull' a diverging contour model toward $\mathbf{x}_0$. Finally, note that after this run the selected particle is that with the highest weighting according to the image and shape priors only, ie. calculated without the attraction force.

In practise, even with the attraction force, the loop closing run may give undesired results if invoked too early, or if the contour was initialised near to a corner in the true boundary. We test the algorithm on extreme cases using a synthetic image (figure 6.6). The image, referred to as the 'heart' image, is an 8-bit grey scale image created by placing a symmetrical heart shape of grey level 128 on a background of grey level 192. The image is then given Gaussian noise with standard deviation 10 and finally smoothed using a 3×3 pixel averaging window.



(a) (b) (c)

Figure 6.6: A loop closing algorithm used on a synthetic image to demonstrate two failing scenarios. (a) The loop is successfully closed by tracking from A to $B = \mathbf{x}_0$. (b) The algorithm is invoked too early and the loop closing run diverges (inset) before re-joining the true boundary. (c) A is close to a sharp corner (inset), which is overshot by the loop closing run.

In figure 6.6 (a) the loop closing algorithm successfully tracks a large boundary section from A to B. In (b) the contour diverges from the true boundary shortly after the start of the loop closing run (inset). The contour eventually heads toward the target due to the attractive force and re-joins the true boundary. In (c) there is a relatively short boundary section between points A and B. However, the section to be tracked is not straight, involving a sharp corner and the contour overshoots the boundary due to the smoothness constraint.

In summary, the loop closing algorithm must be invoked when the subsequent run can realistically hit the target $\mathbf{x}_0$, by tracking a relatively short and straight section of the true boundary. This assumes some common sense on the part of the user, as well as the ability to initialise the contour in a sensible place. In our experience a user gains these intuitions after a modest amount of practice.

6.1.3.2 A method from molecular dynamics

The loop closing algorithm above demonstrates one approach to fixing the end point of a boundary tracker. We now introduce an alternative method inspired by molecular dynamics simulations.

Molecules can be modelled as a chain of atoms (nodes), with rigid bonds (links) of known but variable length. To simulate the dynamics of polymers in a continuum it is often required to generate a large set of random perturbations of a molecular chain. Escobedo and Pablo [271] perturb a molecular chain by removing and 're-growing' a section of the chain between 2 sites that are the origin and target of a random walk. The random walk is constrained so that the final step is bound to reach the target. The number, order, and separation of points between fixed sites are conserved but a new configuration achieved by sequential positioning of nodes after random orientation of the adjoining link (see figures 2-4 in [271]). Angles are selected from a prior distribution that is adapted after each step. Limits are placed on the possible angles to ensure that the distance from the proposed position to the target site does not exceed that of the extreme case given by aligning the remaining nodes in a straight line. The algorithm continues with new limits placed on the prior angular distribution each time. It results that the final step can only take one angle (in fact any angle from a circle on the sphere of 3D angles) so that a step in that direction places the penultimate node exactly one step length away from the target site. The constrained random walk achieves a new configuration inside a homogeneous continuum, i.e. no external energy to influence the final shape. For the purpose of boundary tracking we need to introduce this external energy from image priors.

We have realised boundary tracking with targets, based on the molecular dynamics simulation in [271]. The method is made possible by an inherent flexibility of particle filtering, being that the prediction and weighting stages are independent, so treat internal dynamics and boundary alignment separately. We have designed an adaptive prior angular distribution that constrain 2D random walks, originating at a point A, to reach a known point B. The adaptive prior angular distribution replaces q in the prediction stage of the classical jetstream algorithm. This creates a tracker that is bound to traverse user-defined start and end points whilst still capable of adhering to a region boundary.

At any step of a growing contour, the adaptive angular distribution is constrained to have limits θ_{min} and θ_{max} where θ is the 'global' angle made with respect to the positive x axis. Figure 6.7 shows a schematic diagram of the algorithm at an intermediate point denoted $\{x_1, y_1\}$ between points A and B, to help define variables α, β, D, d and n. The algorithm selects angle φ, equivalent to the *change* in global angle $\Delta\theta$, which must be between the limits θ_{min} and θ_{max} given by.

$$\theta_{\text{min}} = 2\pi - (\beta - \alpha) \qquad \text{and} \qquad \theta_{\text{max}} = \theta_{\text{min}} + 2\beta \tag{6.3}$$

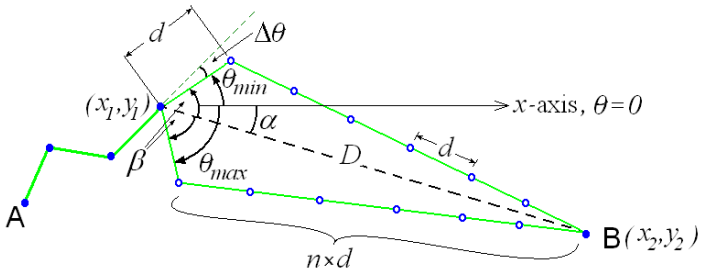


Figure 6.7: Schematic diagram introducing the variables of the tracking algorithm inspired by molecular dynamics. Solid blue dots show previous steps while hollow blue dots represent the steps yet to be taken. There are a fixed number n of steps available after the current step. The dashed line shows the straight line between the current point and the target

where

$$\alpha = \tan^{-1}\left(\frac{x_2 - x_1}{y_2 - y_1}\right) \tag{6.4}$$

by Pythagoras and

$$\beta = \cos^{-1}\left(\frac{D^2 + d^2 - (\min(D + d, nd))^2}{2Dd}\right) \tag{6.5}$$

by the cosine rule, where

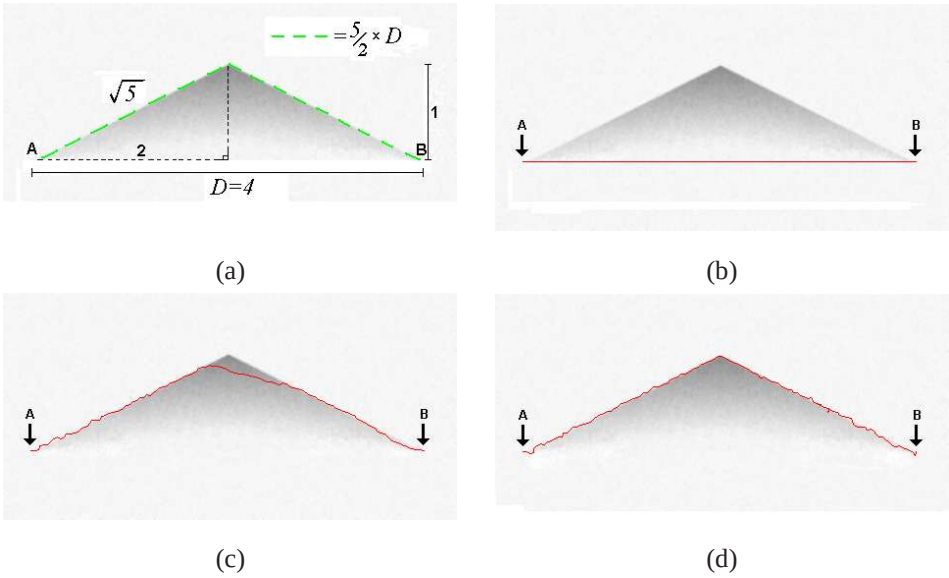
$$D = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{6.6}$$

is the straight line distance from A to B. In this formulation, the prior angular distributions are centred on the direction to the target, not the previous step direction. This means that κ_θ no longer constrains the smoothness of the contour. However, the angle $\Delta\theta$ is known for any given proposal step, so we retrieve smoothness by weighting the proposal steps by $q \times l$ in the familiar way, but have separate angular distributions for prediction $q_p(\theta) = \mathcal{V}(0, \kappa_\theta)$ and weighting $q_w(\Delta\theta) = \mathcal{V}(0, \kappa_{\Delta\theta})$, where κ_θ and $\kappa_{\Delta\theta}$ are separate parameters relating to prediction and weighting.

One problem with this model is that we need to know the future number of steps n between the current proposal step and the target. To overcome this, we allow for an *excess* of available steps, giving rise to the term $\min(D + d, nd)$ in equation 6.5, which returns the smaller of two lengths, either $n \times d$, or $D + d$.

To initialise the algorithm we set the maximum length of the run to be $s \times D$, where s is a *slackness* term, controlling how much longer the contour can be, than the straight line from A to B. To demonstrate the dependence on s we test the algorithm on extreme cases using a synthetic image (figure 6.8). The image, is an 8-bit grey scale image where the greylevels of the trianglular region are graded to remove any discernible edge along the bottom. The image is then given Gaussian noise with standard deviation 10 and finally smoothed using a 3×3 pixel averaging window. The path over the top two sides (dashed green line in 6.8 (a)) has length $\frac{\sqrt{5}}{2} \times D$ where D is the shortest distance from A to B.

When $s = 1$ so that the only possible contour covers the straight line distance to the target, the algorithm produces a straight line from A to B, even though there is no evidence of an edge there in



(a) (b)

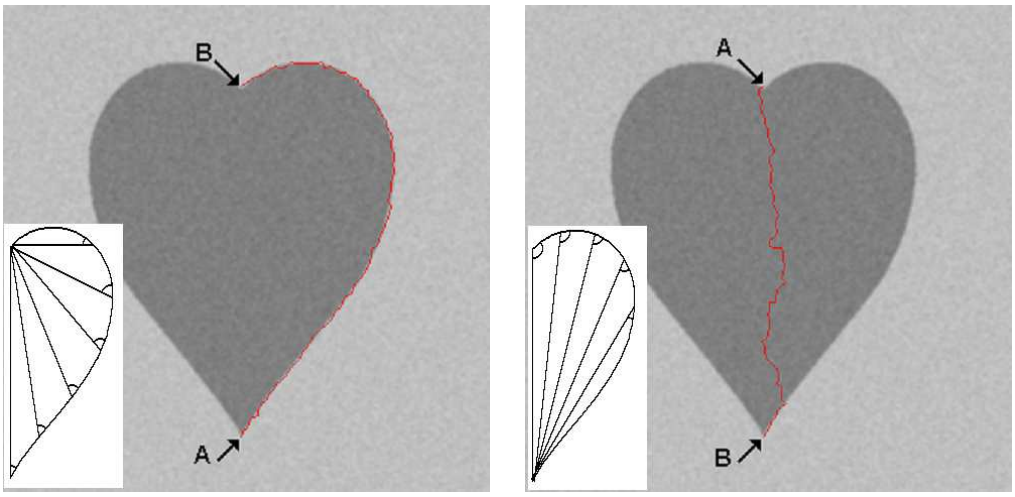
(c) (d)

Figure 6.8: (a) Graded greyscale triangle image with lengths shown (units of 100 pixels). The remaining panels show in red the results of three jetstreams tracking from pixel A to B, with (b) $s = 1$, (c) $s = \frac{\sqrt{5}}{2}$ and (d) $s = 3$.

the image (figure 6.8 (b)). When $s = \frac{\sqrt{5}}{2}$, being the ratio of the true boundary length to the straight line distance from A to B, the whole of the top edge can be extracted in theory. However, afetr the slightest deviation from the true boundary there are not enough remaining steps to return to the true boundary (figure 6.8 (c)). If s is greater than the ratio between the true boundary length and the straight line distance from A to B, the algorithm assumes more steps than are necessary to accurately track the boundary (figure 6.8 (c)). This regime successfully tracks the boundary, and surplus steps are simply discarded when the tracket reaches B.

The balance between κ_θ and $\kappa_{\Delta\theta}$ leads to a directional asymmetry of the tracker, for a given pair of parameters κ_θ and $\kappa_{\Delta\theta}$. Figure 6.9 shows the results of two runs of the algorithm from labelled pixel A to B on the heart image. In both cases the parameters were $s = 5$, $\kappa_\theta = 1.0$ and $\kappa_{\Delta\theta} = 5.0$. In case (a), the angle between the boundary direction and the straight line from the boundary to B is nearly constant and always acute (inset). In case (b) however, these angles are too large for early steps of the tracker (inset). Despite the asymmetry problem, the algorithm can extract large boundary sections between fixed points, as demonstrated in figures 6.8 (d) and 6.9 (a). It is not clear whether the previous algorithm, designed for loop closing, would track such large sections as in the heart example, or handle the corner as in the triangle example.

In summary, the second constrained boundary tracking algorithm, inspired by molecular dynamics simulations, works well in some situations. However, the algorithm uses extra parameters (slackness s and two smoothness parameters κ_θ and $\kappa_{\Delta\theta}$), which must be optimised. The tracker also has an inherent asymmetry which leads to the artefact demonstrated in figure 6.9. For subsequent experiments we choose the first loop closing method, and allow a user to repeat the loop closing run if necessary.



(a) (b)

Figure 6.9: The 'heart' image showing two jetstream runs from pixel A to B, oriented (a) from bottom to top and (b) from top to bottom. *Inset*: diagrams showing how the angle, between boundary direction and a straight line to the target, changes as a tracker progresses from A to B.

6.1.4 Data and performance evaluation

We test the interactive framework to evaluate the main adaptations made to the original jetstream algorithm, ie. the novel interactions of section 6.1.2 and the chosen loop closing algorithm of section 6.1.3.1. We evaluate these adaptations by recording user behaviour during multiple use of the tool to delineate different regions in medical images. This section does not evaluate segmentation quality, which we revisit to compare jetstreams driven by different image models in section 6.1.5.

We test the interactive framework for the chosen application of multiple sclerosis lesion contouring. We choose 3 slices from MR images of different MS patients, and choose 2 lesions in each of these slices. Figure 6.10 shows the 6 regions of interest along with their 'true' boundaries delineated manually by an expert.

6.1.5 Experiments

We asked four experts in MS radiography at London's Institute of Neurology (IoN) to use our interactive jetstream framework for MS lesion segmentation. Each expert rater followed a randomised sequence of 6 lesions, repeated three times for each PD image in figure 6.10, and again for T2 images of the same lesions. In this way, raters completed a total of 36 contours, and repeated this sequence on two separate occasions, making a total of 72 contours.

First, we hypothesise that:

> $\mathcal{H}$6.1.5.1: The loop closing algorithm is
> (a) successful, and
> (b) favoured over a simple, more manual method of completing a contour.

To test this hypothesis we record the acceptance rates and method of completion of all jetstream contours,

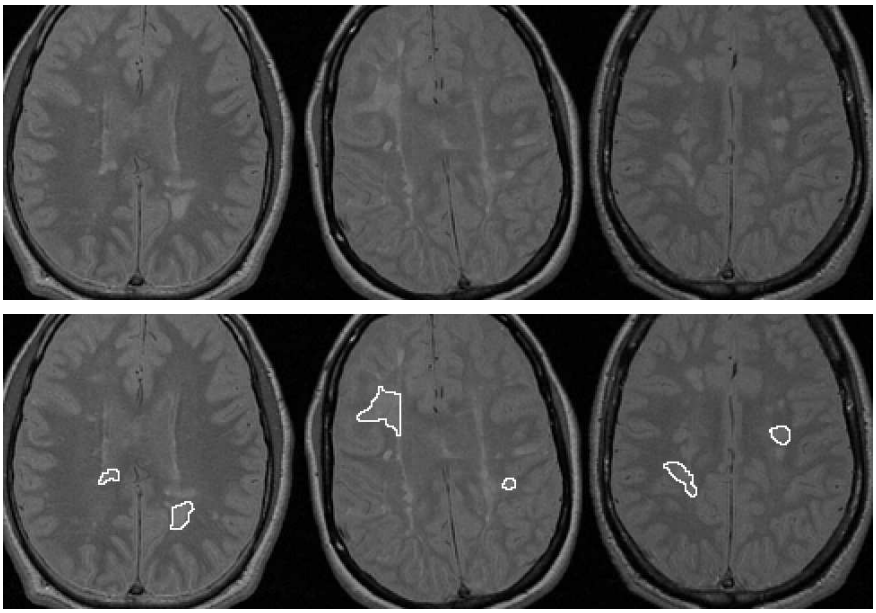


Figure 6.10: *Top row*: PD images of the three chosen axial slices. *Bottom row*: expert delineation of two chosen lesions in each slice.

created by the four raters. We incorporate an 'accept/reject' option such that a user can repeat the final run of the contour if results are undesirable. This reveals whether a loop closing run is considered successful by the expert user. In addition, the user can override the jetstream algorithm and simply close the loop with a straight line. A user might prefer this method of closing the contour, but must already have an almost complete contour so that a short, straight section will lie on the true boundary.

Table 6.1 shows the percentages of loop closing methods/acceptance over all jetstream contours.

Percentage . . .	User 1	User 2	User 3	User 4
. . . finished using loop closer	9.7	95.8	41.7	18.1
. . . of these that only took one attempt	100.0	82.6	46.7	61.5
. . . closed after one attempt using loop closer	9.7	79.2	19.5	11.1

Table 6.1: Level of preference for the loop closing algorithm over a manual method.

When users chose to use the loop closing algorithm, one attempt was usually sufficient (between 46.7 % and 100 % of the time). As a result we accept hypothesis $\mathcal{H}$6.1.5.1 (a). However, table 6.1 reveals large variability in user behaviour. Rater 2 seems to prefer the loop closing algorithm and uses it with a high success rate, while rater 1 usually closed the contours with straight interpolation. We accept hypothesis $\mathcal{H}$6.1.5.1 (b) for rater 2 only. The idiosyncrasy of user behaviour highlights the importance of maximaising user control, not only in terms of steering but in choosing alternative modes of interaction.

Next, we hypothesise that:

$\mathcal{H}$6.1.5.2: The extra interactions of no-go areas, manual particle selection and parameter adjustment are useful in MS lesion contouring.

To test this hypothesis we record the use of extra interactions during segmentation of the 72 lesions. All users had been shown the use of these extra interactions, and had used them at least once before during a practise session.

Apart from a single contour, (drawn on one occasion by a one user), all lesions were segmented without use of the no-go area or manually choosing from alternative paths. In the exceptional case, rater 2 made use of the no-go area *and* three uses of the manual particle selection on the same lesion. We reject $\mathcal{H}$6.1.5.2 in the case of no-go areas and manual particle selection. The rejection of $\mathcal{H}$6.1.5.2 could be due to relative unfamiliarity with the extra image-based interactions. Also, given that any contouring error can be corrected by the anchoring method (figure 6.2) a rater could consider the extra modes of interaction not 'worth' the extra requirement of invoking them.

Parameter adjustment proved to be more useful. Raters generally found a desirable combination of smoothness, detail and length of a jetstream run, by trial and error at the start of a contouring session. Having settled on these parameters, users made adjustments occasionally, such as when they encountered particularly sharp corners. We accept $\mathcal{H}$6.1.5.2 in the case of parameter adjustment.

Next, we hypothesise that:

> $\mathcal{H}$6.1.5.3: Operator time for jetstreams
> (a) is less than freehand drawing, and
> (b) reduces with user experience.

To test these hypotheses we record the number of runs and the absolute time taken to complete a contour. Note that a single run of the jetstream algorithm is completed and displayed in real-time, but segmentation involves successive runs, so the absolute time taken to complete a contour is governed by the number of runs and the length of any pauses between runs. In fact, absolute times are generally longer for jetstreams, so we can not accept $\mathcal{H}$6.1.5.3(a). There are three likely causes of this limitation. First, the raters are still relatively unfamiliar with the tool, as freehand delineation is in common use. Second, raters can, and are known to pause contouring between runs of a jetstream, whereas the freehand tool demands a continuous interaction from start to finish. Third, the application of MS lesion segmentation involves ROIs that are small and jagged, so long boundary sections are unlikely to be tracked by a single run. We suspect that another application, where ROIs are larger and with longer smooth sections of the boundary, would benefit more from the boundary tracker in terms of operator time.

For hypothesis $\mathcal{H}$6.1.5.3(b) we compare operator times on two occasions, contouring the same lesions. We consider that a rater's level of experience is greater during the second contouring session, and perform paired-samples t-tests on the duration of two segmentations of each region, on the first and second occasion. We measure the duration in terms of both absolute time, and the number of jetstream runs N_{runs}, equivalent to the number of anchors placed around a single lesion. Table 6.2 shows the mean durations and p-values indicating significant reductions in user time. We accept hypothesis $\mathcal{H}$6.1.5.3(b) due to the high levels of significance of the reduction in operator time and N_{runs}.

	Occasion	Mean time (sec)	Mean N_{runs}
User 1	first	27.60	18.17
	second	19.07 (p<0.001)	14.89 (p=0.01)
User 2	first	40.17	41.28
	second	28.91 (p=0.025)	34.53 (p=0.1)
User 3	first	44.54	36.75
	second	23.14 (p=0.01)	23.34 (p=0.025)
User 4	first	30.76	31.67
	second	30.76 (p=0.05)	22.39 (p=0.05)

Table 6.2: Mean time and number of runs necessary to complete a given contour on two occasions. P-values indicate the significance of the reduction in user demand.

6.1.6 Conclusions

We have realised an interactive framework for supervised contouring based on jetstreams. User experiments lead to the following conclusions:

- The chosen loop closing algorithm, inspired by probabilistic tractography, is a successful solution to the problem of producing closed contours in a boundary tracking framework.
- Modes of interaction are a matter of user preference, so that in 'giving as complete control as possible to the user' (*Requirement 1*), one form of 'control' is the ability to choose between manual and automated methods.
- Slidebars controlling certain algorithm parameters are generally popular, and allow an algorithm to generalise across user-styles as well as applications, regions within an application and boundary-sections for a given region.
- The methods of on-line supervision between successive jetstream runs leads to a slower tool than simple freehand drawing, but more user experience may make jetstreams faster than the freehand tool.
- The extra interactions and the time-saving potential of the current framework may benefit another application than MS lesions. In particular larger ROIs may better exploit the tracking algorithm.

6.2 SVM Texture Classification

The previous section was concerned with improving boundary tracking by efficient modes of interaction. As well as interactivity we seek an appropriate model of the image data ($\mathcal{C}_2$) to alleviate problems associated with boundary ambiguity. This section develops featureless texture classifiers for the problem of boundary ambiguity and investigates their properties and limitations in synthetic and medical images. We introduce two different classifiers for boundary extraction. The *region-trained* approach seeks to distinguish between textures inside and outside a ROI, which locates the boundary after gradient filtering a

classified image. The *boundary-trained* approach distinguishes boundaries from all other data directly, by treating boundary data as a class of its own.

Section 6.2.1 introduces the data sets used and section 6.2.3 describes how SVMs are evaluated in terms of classification success. Section 6.2.4 uses these data and performance evaluation methods to validate the SVMs and optimise certain model parameters. Experiments in section 6.2.5 address hypotheses regarding the success and generality of the classifiers.

6.2.1 Data and texture sampling

We noted in chapter 2, that the validation and evaluation of new methodologies benefits from the use of synthetic data. Synthetic textures and composite texture images have the advantage over medical data, that the ground truth is known exactly and image properties can be controlled.

6.2.1.1 Synthetic texture images

We use data sets derived from images in the Vision Texture database [44]. These 512×512 images each contain a single 'texture', meaning a continuous scene taken from one semantic class. We choose the textures of 'stone' (figure 6.11 (a)) and 'fabric' (figure 6.11 (d)) as these have finer scale pixel variation compared to the others in the database and in each case, the database provides two different instances allowing us to use different data for training and testing a given classifier. We pre-process the VisTex images in order to emulate properties of medical images and control image contrast using 'histogram specification'. This general technique computes a transform function or 'look-up table' that transforms the image histogram to a specified distribution. We set pixel intensities i to have Gaussian distributions defined by $\exp(\frac{(i-\mu)^2}{\sigma^2})$ and control the first- (μ) and second-order (σ) statistics. We specify the means $\{\mu_{\text{fg}}, \mu_{\text{bg}}\}$ and standard deviations $\{\sigma_{\text{fg}}, \sigma_{\text{bg}}\}$, where subscripts 'fg' and 'bg' denote foreground and background respectively. We choose these statistics to match those of MS lesions and the white matter immediately adjacent to them, for all ground truth in a reference MRI dataset. The reference data is a single MRI volume, and is that having the most representative contrast between region and background. We define this most representative volume as that having the median 'Z-score', which is a standard measure of histogram overlap, given by $Z = (\mu_{\text{fg}} - \mu_{\text{bg}})/\sqrt{(\sigma^2_{\text{fg}}/N) + (\sigma^2_{\text{bg}}/N)}$, where N is the number of ground truth pixels common to both classes. Figure 6.11 (b)/(c) and (e)/(f) shows example histogram transformations and resulting images.

6.2.1.2 Synthetic texture boundaries

We create synthetic data for the boundary-trained SVM by forming composite images from two synthetic textures. We choose two texture images to represent 'foreground' and 'background', and copy their contents inside and outside a synthetic shape. We generate a set of shapes in polar coordinates (r, θ) with origin at the centre of an image, by varying the radius sinusoidally over the range of orientations $\{0 \ldots, 2\pi\}$, along with a random perturbation. We also choose the orientation ($\theta = 0$) and the number of sinusoidal periods at random in the equation

$$r = 120 + 15\text{sin}(M(\theta + \Delta\theta)) + \Delta r, \qquad \theta \in \{0, \frac{2\pi}{70}, \frac{4\pi}{70}, \ldots 2\pi\}, \tag{6.7}$$

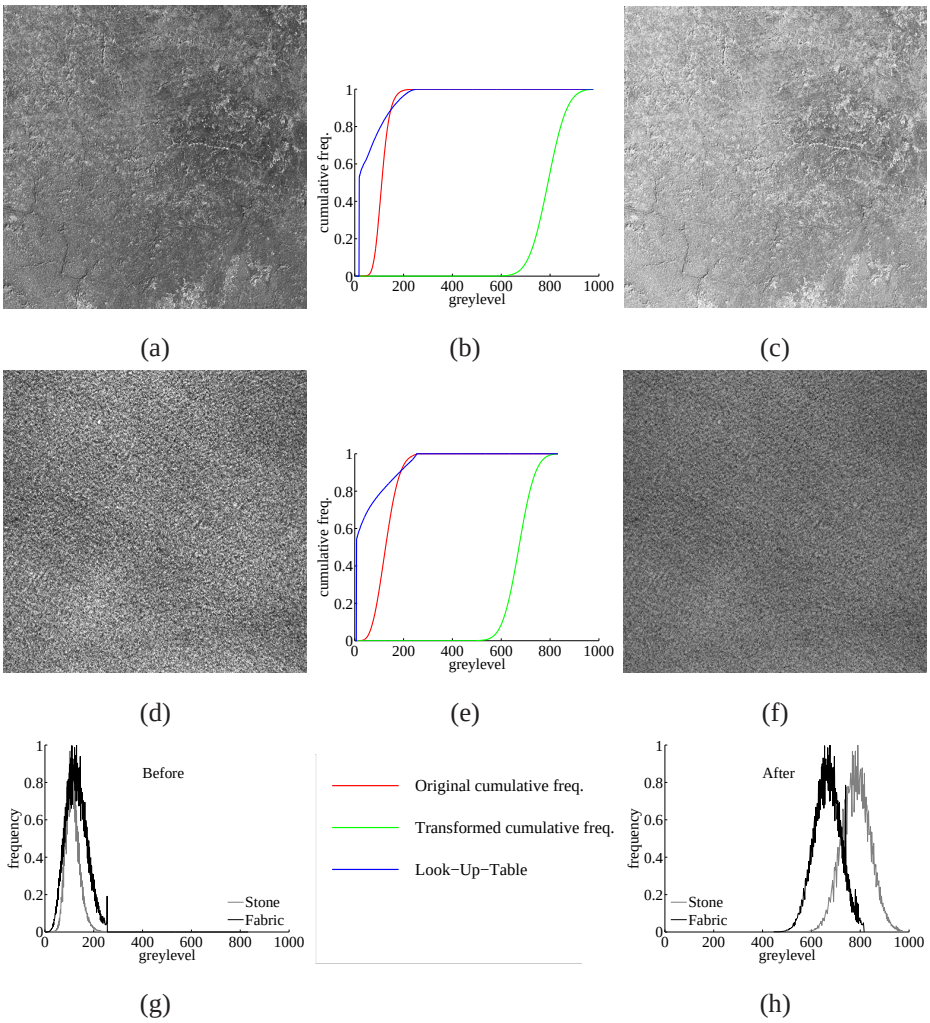


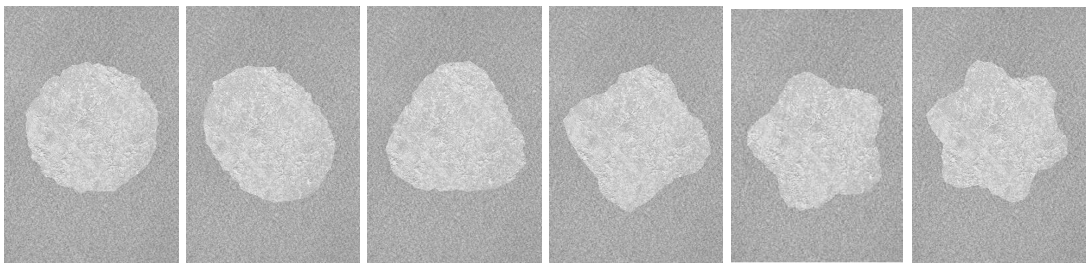
Figure 6.11: Synthetic images derived from the Vision Texture database. (a) Greyscale version of the 'stone' texture. (b) Cumulative histograms of the original intensity distribution and that specified to match MS lesion statistics, shown along with the transform function. (c) Transformed image of 'stone' texture. (d)-(f) the same for 'fabric' texture specified to match background white matter statistics. The overlapping stone and fabric histograms are shown before (g) and after (h) the transformation.

where $\Delta\theta$ is the orientation of the shape selected at random from a uniform distribution in the range $\{0\ldots,2\pi\}$, Δr is the radial perturbation selected from a normal distribution $\mathcal{N}(0,2.5)$ and M is double the number of sinusoidal periods, selected from a uniform distribution over the range $\{2,\ldots,6\}$. The shapes have one boundary point per $\frac{2\pi}{70}$ radians. The 70 points that define each shape are then interpolated by the Bresenham line algorithm.

Figure 6.12 shows composite texture images created using five example shapes. In total we generate 40 shapes, which have a mean of 757 boundary pixels.

6.2.1.3 Medical regions and boundaries

We have MR images of 40 brains containing MS lesions. The datasets are previously labelled by an unknown expert using the tool in [22] along with manual post editing. For each brain there are two 3D



(a) (b) (c) (d) (e) (f)

Figure 6.12: Composite texture images made from synthetic shapes having (a) 1, (b) 2, (c) 3, (d) 4, (e) 5 and (f) 6 sinusoidal periods

data sets, imaging Proton Density (PD) and spin-spin relaxation time (T2) respectively. It is likely that ground truth labelling was performed mainly on the PD slices, with corresponding T2 slices viewed as a reference [19]. MR slices and lesions in figure 6.10 are examples from this dataset.

6.2.2 Sampling for feature vectors

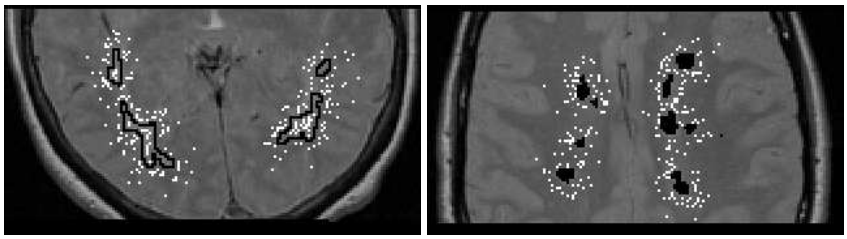
We use the two types of ground truth described above to obtain feature vectors for boundary-trained and region-trained SVMs. In each case, we sample pixels from a neighbourhood centred on each ground truth pixel, and turn the intensities directly into 'feature' vectors.

First, following the region-based classifiers in [203], we use square pixel windows of width w, to sample vectors of dimensionality w^2. Second, for use in the boundary-trained SVM, we use a star-shaped sampling window that spans a width of 7 pixels but gives 25-dimensional feature vectors (inset in figure 6.14). The star-shape creates a 25 dimensional feature vector that spans an area of 7×7 pixels. This sampling window is expected to capture both the smaller scale textures at either side of a boundary and the larger scale texture that a boundary itself comprises [200]. Upon sampling, we re-scale the raw intensity to the range $\{0 \ldots 1\}$ by subtracting the minimum and dividing by the range of all values in the corresponding volume. Such rescaling to a small range improves the performance of SVMs [197].

6.2.2.1 Positive and negative ground truth

SVMs require labelled training data from both positive and negative classes. The two classes are defined differently for the boundary- and region-trained SVMs and for MRI and synthetic images. Figure 6.13 shows labelling schemes for both classifiers in MS lesion data. Positive labels are defined (a) on and (b) inside the ground truth boundaries. For the negative class we label *off-boundary* and *non-lesion* pixels respectively. We select these pixels at random, with probability weighted by $p \propto e^{-\frac{r}{2}}$, where r is the distance to the closest positive class pixel, but reject locations immediately adjacent to positive ground truth to allow for imperfections in the original labelling. Note that in the boundary-trained case, the negative examples include locations inside the lesions.

We define the negative classes in this way so that the SVM discriminates between boundaries and the nearby tissue that a supervised contour model is most likely to encounter. The idea of assigning an exclusive class to data immediately adjacent to a boundary is similar in principle to one of the schemes



(a) (b)

Figure 6.13: Example ground truth pixels (white) for the negative classes of (a) 'off-boundary' and (b) 'non-lesion'. Corresponding positive class pixels are shown in black.

used by de Bruijne *et al.* [119] in their active appearance models. However, our labelling is automatic and the spatial constraint encodes the probabilistic nature of the contour models that we intend to use with the SVMs. The random negative-class pixel labelling terminates when, in a given slice, the same number of negative labels have been assigned as there are positive. This avoids problems that can arise from unbalanced training sets [272].

6.2.3 Performance evaluation

Each SVM assigns distance values d_{SVM} to the feature vectors in a test set. We seek to evaluate classifier performance based on these outputs. Performance evaluation has two roles in this section. First, parameter optimisation in section 6.2.4 is based on maximising classification success. Second, we compare the success of different classifiers in the experiments of section 6.2.5. For both purposes we use the area under an ROC curve (AUC) as a success measure.

The general method of producing ROC curves was described in 4.3 and is explained here more specifically for the SVMs above. We calculate true positive fractions TP and false positive fractions FP by comparing the true labels of test data with the labels assigned by thresholding decision values d_{SVM}. At any threshold, all measurements above the threshold are classified as positives and those below it negatives. Counts of true positives N_{TP}, true negatives N_{TN}, false positives N_{FP} and false negatives N_{FN} give the true-positive fraction TP and false-positive fraction FP by equation 4.8. For all train/test scenarios we keep the number of positive and negative class data the same, and the number of training/testing data the same. In all cases we vary the threshold in 500 increments from the minimum (negative) decision value to the maximum (positive) decision value, and form the ROC curve by plotting FP against TP.

As described in section 4.3, cross-validation such as a 'leave-one-out' scheme removes bias in ROC analysis. For N individual datasets containing ground truth, we train a SVM on data from $(N-1)$ datasets and test it on that data 'left out' of training. By repeating N times with a different set omitted from training each time, we get N values of AUC use the mean AUC as a single performance measure. For this leave-one-out scheme we must first partition the data into distinct sets. In the case of MR images we have 40 separate scan volumes which serves as a meaningful partition. In the case of synthetic images we take random subsets of the available data.

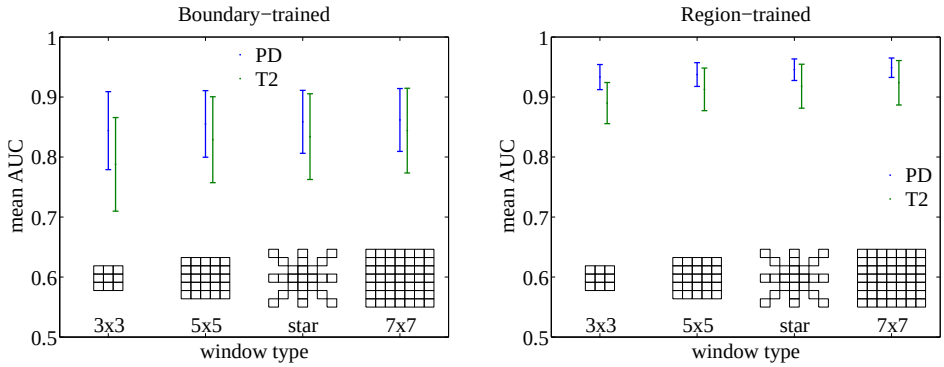
6.2.4 SVM design

We use binary SVM classifiers of the software library 'libsvm' [216], with a radial basis function (RBF) kernel chosen for its ability to generalise across different texture types. In the absence of explicit texture features, the design of a SVM involves choosing texture windows, and investigating the sensitivity to sample sizes and SVM parameters. This section uses ROC analysis to see how SVM performance is affected by changing texture windows, sample sizes and model parameters. We perform these preliminary investigations on both MS lesion data and synthetic textures, which lead to the same conclusions. Only the results for MS lesion data sets are presented here. Unlike later experiments, these preliminary investigations only use a subset of the available MRI volumes in order to satisfy time constraints. We select five volumes at random for use as test data.

6.2.4.1 Texture neighbourhoods

The size of the sampling window used to extract feature vectors should match the scale of discriminating textures present in the ground truth. To observe the value of different sampling windows in MS lesion classification we repeat ROC analyses for SVMs using square sampling windows of 3×3, 5×5 and 7×7 pixels. These result in 9, 25 and 49-dimensional feature vectors respectively. We also construct 25-dimensional feature vectors using the star shaped sampling window described above.

We repeat ROC analysis on the five test data sets and present the mean AUC, with error bars at $\pm$ one standard deviation, for the various classifiers. Figure 6.14 shows the results for the various window types and the inset diagrams show the windows. Figure 6.14 suggests that classifier performance increases with



(a) (b)

Figure 6.14: Plots of classifier performance for different sampling windows. Results for PD and T2 lesions are shown for (a) the boundary-trained SVM and (b) the region-trained SVM. *Inset*: diagrams of the sampling windows.

the dimensionality of input feature vectors. This is intuitive, as larger texture patches capture texture at larger scales *in addition* to the smaller scale textures closer to the centre of the patch. Although the 49-dimensional (7×7 window) SVM may perform slightly better, we choose to use 25-dimensional feature vectors for the following experiments to satisfy time constraints. The 5×5 square window and the star-shaped sampling window both form 25-dimensional feature spaces for the respective SVMs,

but the star-shape window spans a larger area in the image. Although there is no significant difference in performance between these two classifiers, we choose the star-shaped window for boundary-trained SVMs by intuition, as these SVMs need to capture texture at either side of a boundary and would be more affected by the inaccuracy of ground truth.

6.2.4.2 Training sample size

The size of a training set affects the ability of a SVM to capture discriminating textures and generalise well across all unseen examples. To observe the affect of training set size we first choose an image that will be used in testing, which is removed from the training set. Next, we group together all available ground truth in the remaining 39 images and a subset of the pooled ground truth selected at random. The subsets are defined as 0.1%, 0.5%, 1%, 5%, 10%, 50% and 100% of the pooled ground truth. We repeat ROC analysis for a pre-selected set of five test images and calculate the mean AUC, for each of the training sizes. Results suggests that there is little to be gained from using more than 50% of the available ground truth as training examples, for both image types and both SVM types. We use this subset (50%) in the following two experiments to allow the desired amount of training and testing in a reasonable time.

6.2.4.3 SVM parameters

We investigate the effect of varying RBF kernel width γ and the cost of 'slack variables' c using a grid search to determine which (γ, c) *pair* gives optimal SVM performance. Preliminary investigations suggest that our SVMs are not very sensitive to c while γ values should be set around unity. Based on the scheme given in [197] and used in [195], we vary each parameter exponentially. Figure 6.15 shows the topology of AUC vs γ and c for the various classifiers. Based on these preliminary results we choose to use γ = 1.0 and c = 10 in the following experiments. A more exhaustive parameter search is desirable, but was impossible prior to the following experiments due to time constraints.

6.2.5 Experiments

This section evaluates the performance of various texture SVMs using ROC analysis.

6.2.5.1 Comparing texture and intensity

First, we hypothesise that:

> $\mathcal{H}$6.2.5.1: Texture SVMs are an improvement over intensity thresholding for discriminating regions and boundaries.

To test this hypothesis, we repeat ROC analysis by thresholding both the output of SVM classifiers d_{SVM} and image intensities. We repeat for synthetic and MS lesion textures.

In the case of MS lesions we use all available ground truth data. This means that the leave-one-out scheme is repeated for 40 testing brains, and in each case all of the ground truth from the 39 remaining brains are used in training. This leads to a mean and standard deviation calculated over 40 AUC values.

We repeat for region- and boundary-trained SVMs and for T2 and PD images. In the case of synthetic textures, we use 7500 training/testing vectors, chosen at random from each image. We use one pair of stone and fabric images for training and a second pair for testing. We finally partition the

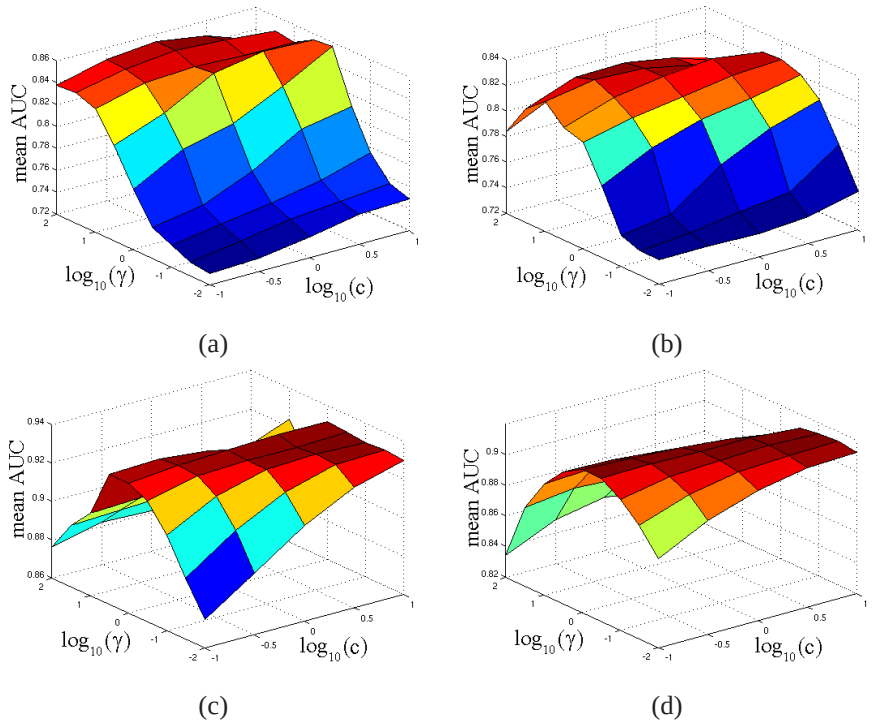


Figure 6.15: Surface plots of classifier performance for varying γ and c. The top row shows results for the boundary-trained SVM classifying (a) PD and (b) T2 images. The bottom row shows results for the region-trained SVM classifying (c) PD and (d) T2 images.

training and testing sets into 10 subsets, by sampling random locations from the images, to perform cross-validation.

For intensity thresholding in MRI data sets we perform two different ROC analyses, with different interpretations. In one case, we construct ROC curves *per brain*. This means that we threshold the ground truth in each brain to produce 40 separate curves, then take the mean and standard deviation of the AUC values. The *per brain* analysis tells us how well the ROIs in a given image contrast against the surrounding tissue. In another case, we group all the ground truth together from the 40 brains and vary the intensity threshold to produce a single *global* ROC curve.

For all intensity ROC analyses, the negative class ground truth is represented by the same, localised random selections as used for SVM training. Table 6.3 shows the mean AUC ($\pm$ one standard deviation) for the two SVMs, along with results for intensity thresholding. In the case of synthetic images we do not perform intensity thresholding on 'boundary' ground truth, because the boundary is defined between adjacent pixels, so all single pixels are either inside or outside a region. These results suggest that SVM texture is an improvement over PD or T2 intensity for discriminating MS lesion ROIs and boundaries. We perform independent t-tests for the MS lesion results to quantify the significance of this improvement. In the case of *per brain* intensity analysis, tests reveal that, for T2 data, the improvement of both region- and boundary-trained SVMs is significant with a confidence of $> 99.95\%$. For PD data, boundary-trained SVMs classify significantly better with a confidence between 90% and 95%, whereas the improvement of the region-trained SVMs is not significant (confidence $< 90\%$). Comparing with *global* intensity

	Region ground truth			Boundary ground truth		
Image type	SVM texture	Intensity (per brain)	Intensity (global)	SVM texture	Intensity (per brain)	Intensity (global)
PD	0.928±0.055	0.924±0.020	0.836	0.875±0.043	0.705±0.062	0.610
T2	0.927±0.021	0.881±0.032	0.854	0.858±0.039	0.687±0.071	0.624
Synth.	1.00± <0.001	N/A	0.93±0.003	0.95±0.003	N/A	N/A

Table 6.3: Comparison of mean AUC (± one standard deviation) for SVM texture classification and intensity thresholding at regions and boundaries.

analysis, SVMs are a significant improvement in all cases.

We accept hypothesis $\mathcal{H}$6.2.5.1, but note that the improvement of the region-trained SVM over the per-brain intensity thresholding is not significant, and may be negligible for a given MR image.

6.2.5.2 Locally trained SVMs

In the case of MS lesion textures we also investigate the use of *Locally trained* SVMs. These classifiers are trained using ground truth from one slice only, and used to classify data in an adjacent slice. Such a classification scheme has two possible uses in a segmentation framework, which draw from the incremental learning literature reviewed in chapter 4. First, training data could be accumulated during run-time, as the user identifies more ground truth interactively. Second, 3-dimensional segmentation tasks involving contouring of successive slices could use the segmentation accepted in one slice to train a classifier for the next.

A locally trained SVM is confounded by having a small training set, but at the same time benefits from the coherence of training and testing data. We hypothesise that:

$\mathcal{H}$6.2.5.2: Locally-trained SVMs perform at least as well as globally trained SVMs.

To test this hypothesis, we train a *one-slice-SVM* on ground truth from all lesions in one slice, and test it on all lesions in an adjacent slice. We repeat by swapping the training/testing sources, and for slice pairs from three different brains shown in figure 6.16. We then take the mean and standard deviation of AUC values over the six training/testing cases for region- and boundary-trained SVMs and for both image types. This allows independent t-tests to look for significant differences between the performance of locally-trained SVMs and those using all available ground truth as in table 6.3. For each slice pair the two AUC values and their mean are given in table 6.4, where 'trained on A' implies tested on B and vice versa, and the image names 'p#s2' correspond to the axial slices shown in figure 6.16. Table 6.4 suggests that, compared with the global case in table 6.3, locally-trained SVMs perform at least as well on region data but less well on boundaries. An independent t-test reveals that the boundary-trained classifiers in table 6.3 perform significantly better than the locally-trained counterpart.

We accept hypothesis $\mathcal{H}$6.2.5.2 for region-trained SVMs but reject it for boundary-trained SVMs. The rejection of $\mathcal{H}$6.2.5.2 for local boundary-trained SVMs could be due to the fact that there are generally less ground truth pixels making up a boundary than are contained inside the region. At the same

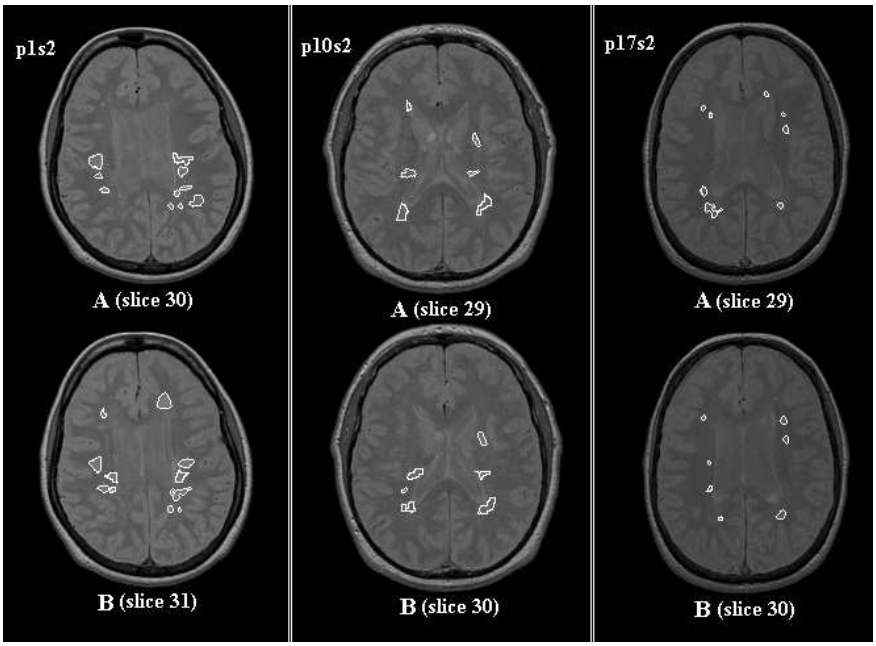


Figure 6.16: PD images of the three chosen pairs of adjacent axial slices. The images in each pair are labelled A and B, and their slice numbers given.

	Boundary-trained SVMs			Region-trained SVMs		
Image/type	trained on A	trained on B	mean	trained on A	trained on B	mean
p1s2 PD	0.815	0.789	0.802	0.963	0.951	0.957
p10s2 PD	0.713	0.744	0.723	0.912	0.894	0.903
p17s2 PD	0.785	0.797	0.791	0.950	0.953	0.952
PD	Overall mean 0.774±0.038			Overall mean 0.937±0.028		
p1s2 T2	0.834	0.807	0.821	0.958	0.909	0.934
p10s2 T2	0.812	0.740	0.776	0.951	0.911	0.931
p17s2 T2	0.788	0.719	0.754	0.972	0.920	0.946
T2	Overall mean 0.783±0.045			Overall mean 0.937±0.027		

Table 6.4: Classifier performance of small, locally trained SVMs. Image names correspond to the axial slices shown in figure 6.16.

time, it is likely that boundaries require more ground truth than regions, to achieve the virtual rotational invariance described in section 4.2.1 and to overcome the inaccuracy of ground truth.

The acceptance of $\mathcal{H}$6.2.5.2 for local region-trained SVMs motivates the use of region-data in slice-by-slice incremental learning. For an idea of whether such classifiers could be exploited in real time, we observe training and testing times for the locally-trained SVMs. Mean training times were $0.24s$ and $0.22s$ for boundary- and region-trained SVMs respectively. The respective mean testing times were $0.059s$ and $0.075s$. Such short timescales suggest that some form of training and testing could be used

during run-time without causing noticeable delay to the segmentation process. However, only the ground truth pixels are classified in the testing stages above, whereas in practice all of the (masked) brain tissue in a slice would be classified.

6.2.6 Conclusions

This section has developed SVM classifiers for discriminating regions and boundaries in textured images. After optimising the classifier design we arrive at the following conclusions:

- For synthetic textures and MS lesions, the featureless SVMs give good classifier performance.
- The approach is capable of locating both regions and boundaries.
- ROC analysis supports the use of both region- and boundary-trained SVMs in segmentation algorithms.
- The method extends to other applications, as it does not rely on multispectral data or application-specific pre-processing.
- Locally-trained SVMs classify *regions* at least as well as those trained across multiple datasets.
- Locally trained SVMs lend themselves to incremental learning schemes, that can benefit a segmentation tool either by responding to interactions or propagating learned image priors through image slices.

6.3 SVM Jetstreams

Experiments above motivate the use of region- and boundary-trained SVMs as image models ($\mathcal{C}_2$) to alleviate the problem of boundary ambiguity in supervised segmentation. The exception, revealed by the test of hypothesis $\mathcal{H}$6.2.5.1, is that the benefits of region-trained SVM over intensity thresholding in a single image is not significant. However, we did see some improvement and this may still benefit a segmentation tool, given that the results of the classifier are first high-pass filtered and it is the gradient magnitude of the decision value that drives the contour model. As such, we are motivated to test the benefit of both SVMs in segmentation.

This section introduces a framework for supervised ROI contouring that combines the boundary tracking algorithm in section 6.1 with image models based on the SVMs in section 6.2. We use both boundary- and region-trained SVMs to drive the adapted jetstreams. In the first case, the distance to the hyperplane output by the boundary-trained SVM gives a measure of *boundariness*. This decision value, denoted d_b, replaces the intensity gradient magnitude. We separately infer local boundary direction by convolving the image of d_b with two orthogonal 'ridge' templates to give the components of a direction vector R_x and R_y, and approximate the local boundary direction by $\tan^{-1}(\frac{R_y}{R_x})$.

In the second case, classification by the region-trained SVM yields the distance value d_r. We use directional Sobel filters to calculate the gradient vector $\mathbf{g}_{d_r}$ of the classified image, which has components $g_{d_r x}$ and $g_{d_r x}$ in the x and y directions. The magnitude of this vector gives a measure of boundariness and we calculate local boundary direction by $\tan^{-1}(\frac{g_{d_r y}}{g_{d_r x}})$.

We classify the whole of a slice in an off-line step. Figures 6.17 and 6.18 show synthetic and MS lesion images along with the magnitudes of (a) intensity gradient, (b) gradient of the images classified by the region-trained SVMs and (c) the output from boundary-trained SVMs.

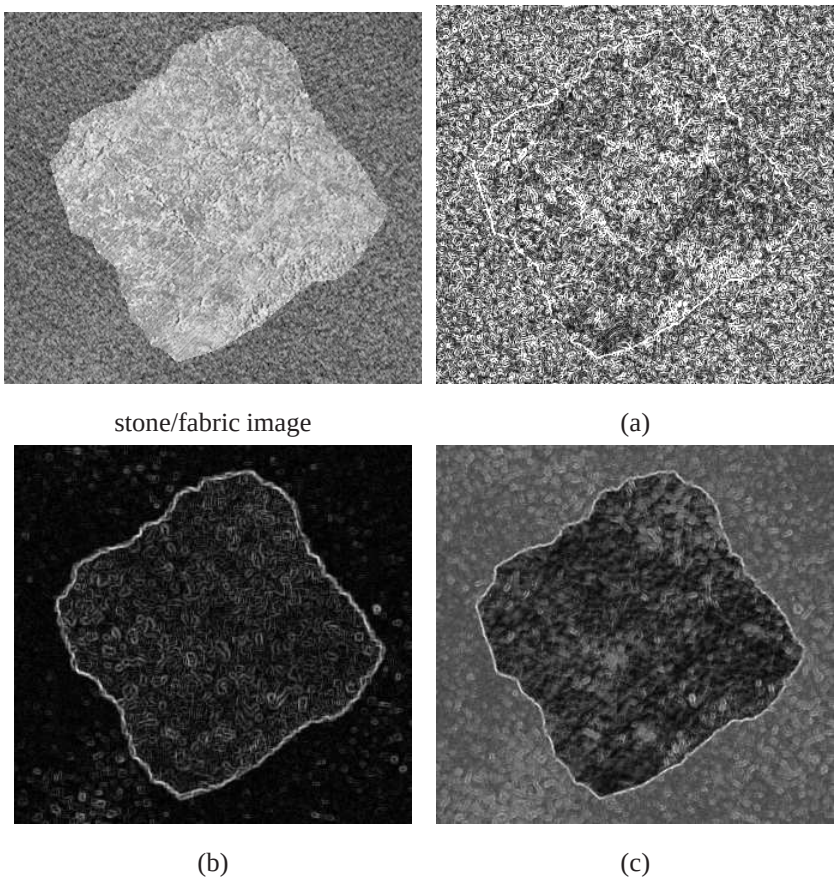


Figure 6.17: *(Top left)*: synthetic image of stone (foreground) and fabric (background) textures. (a) Magnitude of intensity gradient. (b) Magnitude of the gradient of d_r, (c) Raw d_b values.

6.3.1 Performance evaluation

To evaluate a segmentation framework, we must make choices regarding both a performance measure and a comparison method. These two choices should be made to suit the application and the specific component of the segmentation framework that is under investigation (the variable). In this case the application is supervised region contouring and the variable is the image model ($\mathcal{C}_2$).

We evaluate the performance of a tool by measuring accuracy and inter/intra-operator variability. Accuracy refers to the agreement of any contour with the 'true' region boundary. For synthetic images we know the ground truth exactly, whereas in medical images, a true segmentation is not available. To assess any semiautomatic segmentation we choose to use the freehand delineation of the corresponding ROI, as drawn by the corresponding operator, as the ground truth. Given that a user has ultimate control over both jetstreams and manual delineation, we assume that the two results would be the same if jetstreams were perfectly 'accurate'. However, by the same assumption, two manual segmentations of the same region, drawn by the same user, would be identical, which is never the case. As such we assess jetstreams in

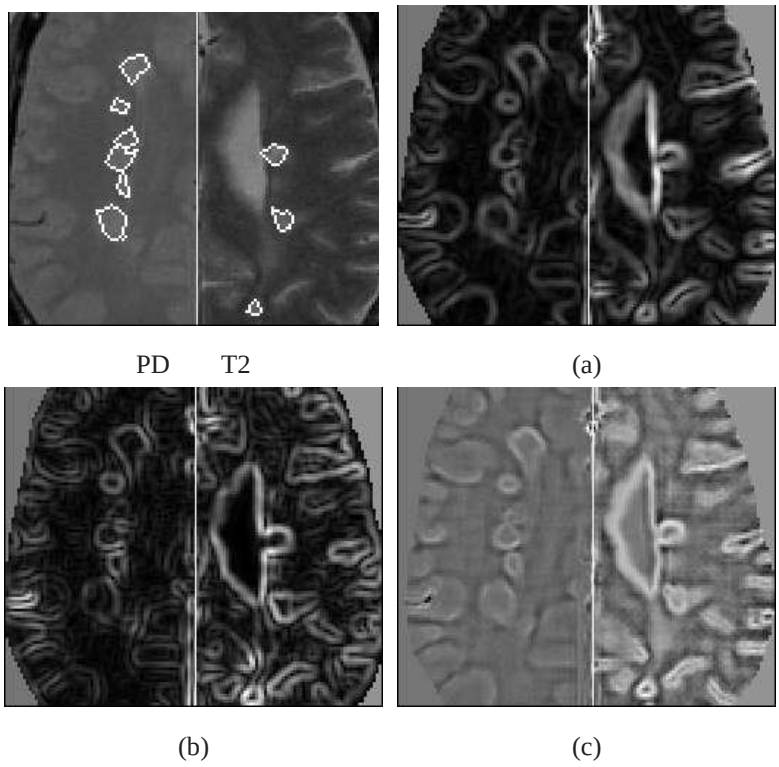


Figure 6.18: *(Top left)*: axial MRI slice showing PD (left hemisphere) and T2 (right hemisphere) intensity. White contours show the ground-truth segmentation of lesions. (a) Magnitude of intensity gradient. (b) Magnitude of the gradient of d_r, (c) Raw d_b values.

terms of the *relative* accuracy compared with the agreement of two manual contours.

Inter-operator variability, or 'repeatability', refers to the agreement between contours produced by different users for the same ROI. Any disagreement might reflect an intrinsic difference in opinion between two raters. However, where two users perceive a region in the same way, this variability is minimised by a good segmentation tool. Intra-operator variability, or 'precision', refers to the agreement between contours produced for one ROI by the same user at different times. This can reflect inevitable human error, but again can be reduced by good software.

We measure the 'agreement' between any two contours using one boundary-based and one region-based similarity measure. A boundary-based measure is arguably more relevant to a boundary-based segmentation tool such as boundary tracking. On the other hand, a region-based measure is relevant to the practical purpose of a segmentation tool, as in the case of MS lesion contouring where a key derivative of segmented images is the 'lesion load', calculated from the area inside lesions.

To measure boundary-based similarity we find the distance from each point on one contour to the closest point on the other and take the average of these distances. This gives the mean minimum distance (MMD), equivalent to the modified Hausdorff distance of [107], where a lower value indicates higher similarity. To measure region-based similarity we use the Dice Similarity Coefficient [104] (DSC), with values ranging from 0 (no overlap) to 1 (perfect overlap) given by DSC $= \frac{2N(A \cap B)}{N(A)+N(B)}$, where $N(A)$ denotes the number of pixels in region A and so on.

Next, we choose which segmentation methods to compare. This choice should isolate the relevant components of the segmentation framework for evaluation. We are concerned with the benefits of the texture classifiers and the effectiveness of the modes of interaction in guiding a contour round a ROI boundary. First we compare SVM jetstreams with the same tool driven by the intensity gradient. We also compare the variability of jetstreams with that of the freehand tool, as freehand drawing and jetstreams are used in a similar way, i.e. guiding an open contour around the whole of a closed boundary.

We test for significance of the differences in performance between two contouring methods using paired t-tests, where a pair refers to the two contouring methods. In the case of accuracy and intra-operator variability and there are six pairs arising from the six lesions involved, and we perform separate t-test for each user. In the case of inter-operator variability, for P users there are $\sum_{i=1}^{P-1} i$ unique *pairs* of users and we perform separate t-test for each region.

6.3.2 Experiments with synthetic texture regions

We ask 5 volunteers to segment the 6 synthetic ROIs in figure 6.12. In all cases a user segments each ROI on 2 separate occasions, and on each occasion using 4 methods. The methods are jetstreams driven by region-/boundary-trained SVMs and intensity gradient, and a free-hand tool. We evaluate the segmentation accuracy and intra-/inter-operator variability of all methods using boundary- and region-based similarity measures.

6.3.2.1 Accuracy

First, we hypothesise that:

> $\mathcal{H}$6.3.2.1: SVM jetstreams used to segment synthetic texture regions are
>
> (a) more accurate than the intensity-driven jetstreams, and
>
> (b) more accurate than freehand delineation

To test hypothesis $\mathcal{H}$6.3.2.1 we measure the accuracy of each method, by the similarity between jetstream contours and the exact ground truth, and take the mean over all ROIs. Figure 6.19 shows the results separately for each user. T-tests reveal that there is no significant difference between the accuracy of the three jetstreams. However, all three jetstreams are significantly more accurate than freehand segmentation for three out of five users (user 2, 4 and 5). One user (3) uses the freehand tool with significantly higher accuracy than the jetstream driven by the region-trained SVM in terms of DSC alone. We reject hypothesis $\mathcal{H}$6.3.2.1 (a) and accept $\mathcal{H}$6.3.2.1 (b) for the majority of users.

6.3.2.2 Intra-operator variability

Next, we hypothesise that:

> $\mathcal{H}$6.3.2.2: in terms of intra-operator variability, SVM jetstreams used to segment synthetic texture regions are
>
> (a) better than the intensity-driven jetstreams, and
>
> (b) better than freehand delineation

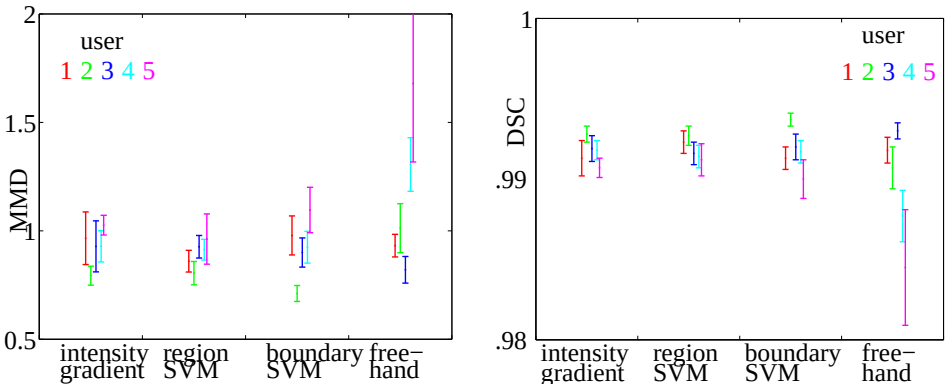


Figure 6.19: Accuracy of jetstreams used in synthetic images, measured by (a) mean minimum distance and (b) Dice similarity coefficient. Error bars are given at $\pm$ 1 standard deviation.

To test hypothesis $\mathcal{H}$6.3.2.2 we measure the intra-operator variability of each method, by the similarity between contours created to segment the same region on two occasions, and take the mean over all ROIs. Figure 6.20 shows the results separately for each user. Independent t-tests reveal that the only

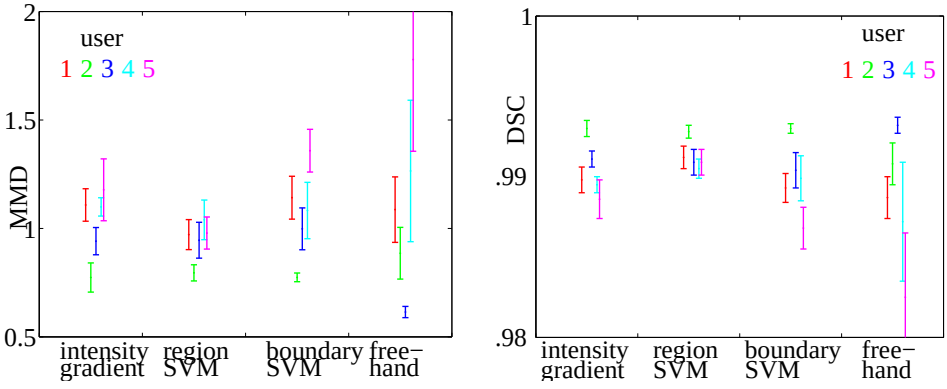


Figure 6.20: Intra-operator variability of jetstreams used for synthetic images, measured by (a) mean minimum distance and (b) Dice similarity coefficient.

significant difference between the three jetstream types is between the two SVM jetstreams, whereby the variability in terms of DSC was significantly lower for the region-trained SVM than for the boundary-trained SVM in the case of user 1. As such we reject hypothesis $\mathcal{H}$6.3.2.2 (a).

The trends in figure 6.20 suggest that the SVM jetstreams give generally better and less varied results than freehand segmentation. For 3 out of 5 users, at least one jetstream gave significantly lower variability than the freehand tool in terms of one or both of DSC and MMD. This is true for all jetstreams in the case of users 2 and 5, and the tool driven by region-trained SVM in the case of user 1. However, all jetstreams showed no significant benefits over freehand segmentation for user 4 and significantly *higher* variability for user 3. As such we can not accept hypothesis $\mathcal{H}$6.3.2.2 (b), but suggest evidence for it

based on figure 6.20, and assume that the improvement would become more apparent after more practice on the part of the user.

6.3.2.3 Inter-operator variability

Next, we hypothesise that:

> $\mathcal{H}$6.3.2.3: in terms of inter-operator variability, SVM jetstreams used to segment synthetic texture regions are
>
> (a) better than the intensity-driven jetstreams, and
>
> (b) better than freehand delineation

To test hypothesis $\mathcal{H}$6.3.2.3 we measure the inter-operator variability of each method, by the similarity between contours created to segment the same region by four users, and take the mean over all 10 unique pairs of users. Figure 6.21 shows the results separately for each region.

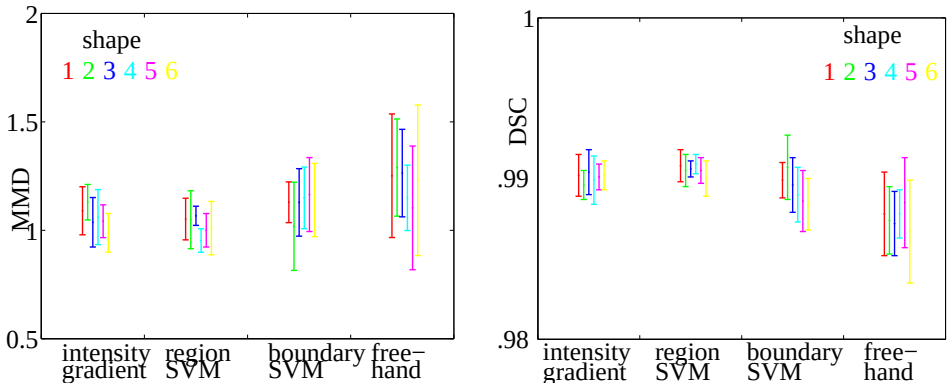


Figure 6.21: Inter-operator variability of jetstreams used for synthetic images, measured by (a) mean minimum distance and (b) Dice similarity coefficient.

T-tests reveal no significant difference between SVM jetstreams and the intensity-driven tool, so we can not accept hypothesis $\mathcal{H}$6.3.2.3 (a). However, the trend in figure 6.21 suggests lower variability for at least the region-trained SVM. Indeed, the region-trained SVM shows some significant improvement over the boundary-trained SVM. This is true for region (iii) in terms of MMD and (iv) in terms of both measures. In terms of DSC, the region-trained SVM significantly out-performs freehand segmentation in all cases, with the boundary-trained SVM also showing significant improvement for regions (i) and (ii). The region-trained SVM also gives significant improvement in terms of MMD for regions (i) and (iii). We accept hypothesis $\mathcal{H}$6.3.2.3 (b).

6.3.2.4 Useability

The final hypothesis concerns the usability of SVM jetstreams. We hypothesise that:

> $\mathcal{H}$6.3.2.4: SVM jetstreams used to segment synthetic texture regions place less demand on the user than jetstreams driven by intensity gradient.

To test hypothesis $\mathcal{H}$6.3.2.4 we count the number of anchors used to complete a closed contour using jetstreams driven by each of the three image models. We take the mean over 6 regions and look for significant differences between the SVM driven by intensity gradient and each of the SVM types. Table 6.5 shows the results for each user. Bold numbers with superscripts '+' and '-' respectively, denote tools that were significantly more and less user friendly, in terms of demand, than the jetstream driven by intensity gradient. In all cases, use of the region-trained SVM leads to less user demand than using

Image model	User 1	User 2	User 3	User 4	User 5
intensity grad.	55.67±10.13	64.33±10.46	71.17±6.43	54.50±9.59	37.83±7.60
region SVM	**31.67±3.33**$^{+}$	55.33±6.98	65.00±9.84	**29.67±4.84**$^{+}$	**18.17±4.22**$^{+}$
boundary SVM	54.83±15.89	82.67±25.85	**83.83±9.77**$^{-}$	**70.17±13.06**$^{-}$	38.67±6.95

Table 6.5: Mean number of anchors required to segment synthetic texture regions. Bold numbers denote a significant difference between jetstreams driven by SVM and intensity gradient ($p \leq 0.05$).

intensity gradient and this improvement is significant for the majority of users. However, use of the boundary-trained SVM leads to a reduction in useability for the majority of users, which is significant in two cases. We accept hypothesis $\mathcal{H}$6.3.2.4 for the region-trained SVM but reject it for the boundary-trained SVM.

6.3.3 Experiments in MS lesion contouring

The following experiments test the efficacy of the SVM jetstreams for the application of MS lesion contouring. In order to test the tool in a realistic setting we ask expert raters to perform segmentations. These are 4 trained raters with experience of MS lesion contouring who, at the time of the experiments, work at the Institute of Neurology (IoN), London. The experts segment MS lesions using three jetstreams as above, and the freehand drawing in the experimental protocol described in section 6.1.5, where three repeated segmentations of the same lesion refer to the use of jetstreams driven by the three different image models. All raters also segmented each of the 6 lesions using a freehand tool. In summary, on a single occasion, each rater followed a randomised sequence of 24 tasks from 6 lesions and 4 methods. We use the results to compare the texture-driven jetstreams with intensity-driven jetstream and the freehand tool.

6.3.3.1 Accuracy

First, we hypothesise that:

> $\mathcal{H}$6.3.3.1: SVM jetstreams used to segment MS lesions are
> (a) more accurate than the intensity-driven jetstreams and
> (b) at least as accurate as freehand delineation

To test hypothesis $\mathcal{H}$6.3.3.1 we must measure the accuracy of contours with respect to some definition of 'ground truth'. Unlike the synthetic images above, true boundaries are not defined for the MS lesions. To measure the accuracy of each jetstream, we take the similarity between contours created by

a rater using that jetstream and the same rater's freehand contour. To quantify freehand accuracy we measure the similarity between a freehand contour and a second one created by the same rater, for the same ROI, days earlier. Figure 6.22 compares the mean similarity over all 6 lesions, of each method used on PD and T2 images. T-tests reveal no significant difference between the three SVMs for either PD or

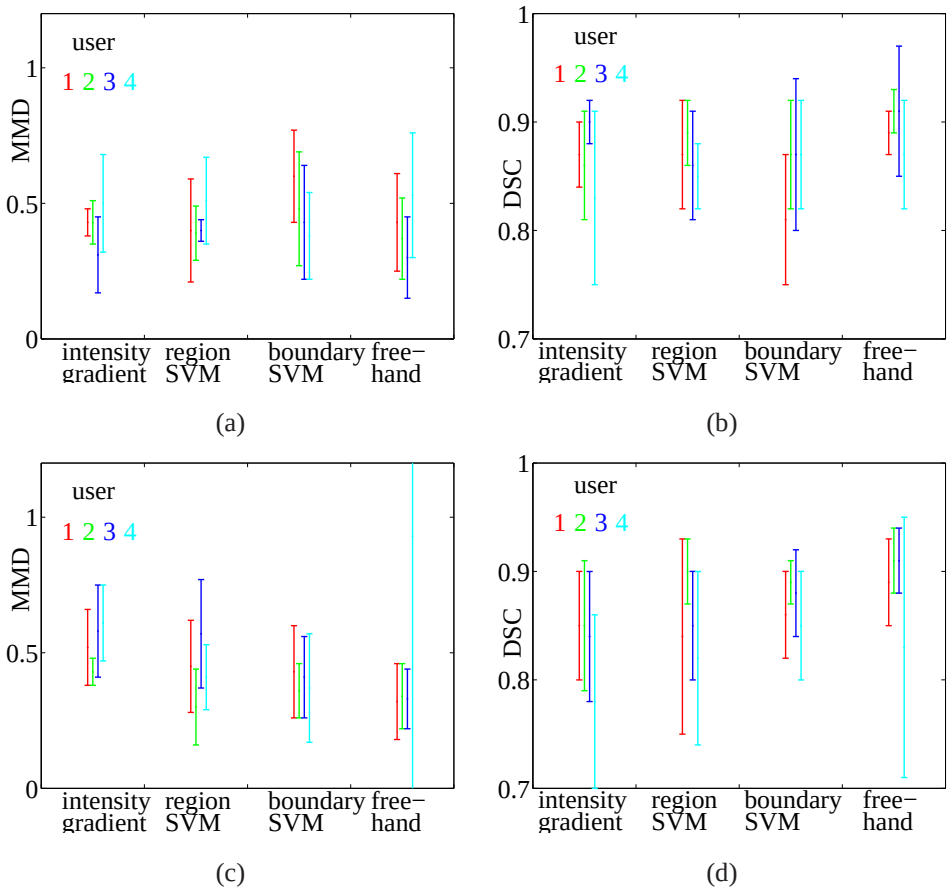


Figure 6.22: Accuracy of jetstreams used for MS lesion images. *Top row:* for PD images by (a) mean minimum distance and (b) Dice similarity coefficient. *Bottom row:* for T2 images by (c) mean minimum distance and (d) Dice similarity coefficient.

T2 images. The only significant difference between jetstreams and freehand tool is for the case of user 1, who achieved higher accuracy with the freehand tool in PD images. We can not accept hypothesis $\mathcal{H}$6.3.3.1.

6.3.3.2 Intra-operator variability

Next, we hypothesise that:

> $\mathcal{H}$6.3.3.2: in terms of intra-operator variability, SVM jetstreams used to segment MS lesions are
>
> (a) better than the intensity-driven jetstreams, and
>
> (b) better than freehand delineation

To test hypothesis $\mathcal{H}$6.3.3.2 we measure the intra-operator variability of each method, by the similarity between contours created to segment the same region on two occasions, and take the mean over all ROIs. Figure 6.23 shows the results separately for each user. For PD regions, T-tests reveal no significant

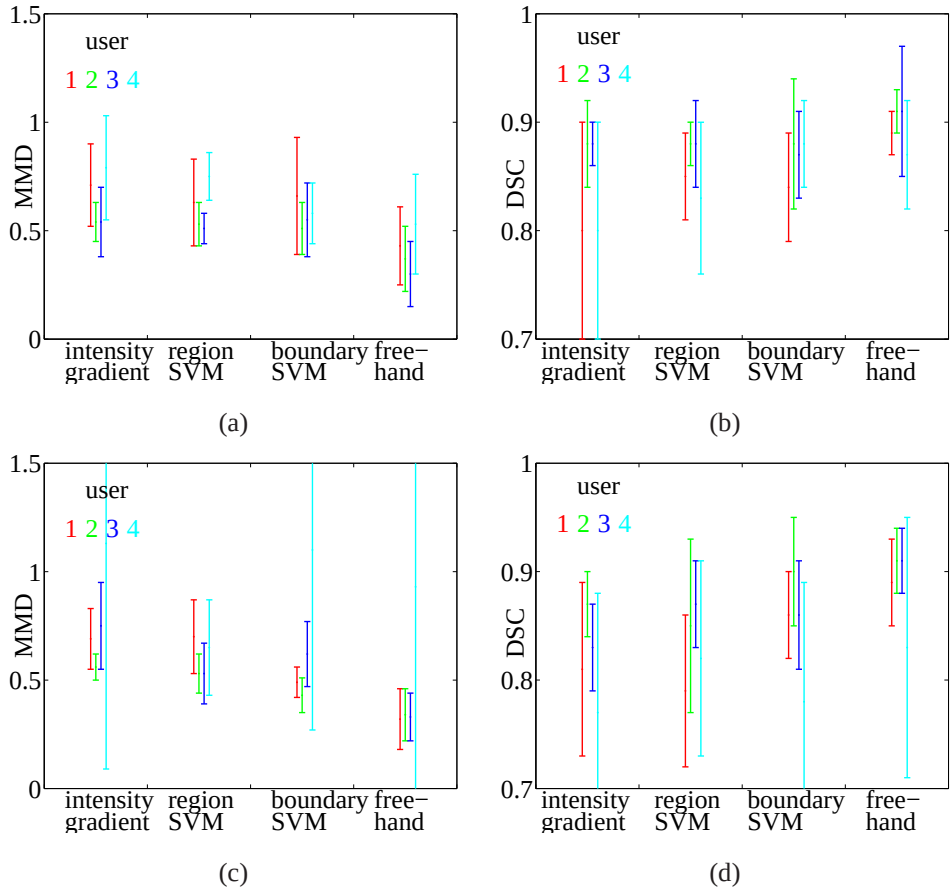


Figure 6.23: Intra-operator variability of jetstreams used for MS lesion images. *Top row:* for PD images by (a) mean minimum distance and (b) Dice similarity coefficient. *Bottom row:* for T2 images by (c) mean minimum distance and (d) Dice similarity coefficient.

differences between jetstreams or with respect to freehand segmentation, in terms of either similarity measure. Similarly for T2 regions there was no significant difference in terms of DSC, while in terms of MMD, SVM jetstreams performed significantly worse in some cases. This is seen for the region-trained SVM (users 1 and 2) and boundary-trained SVM (user 3). We can not accept hypothesis $\mathcal{H}$6.3.3.2 based on the current data, but expect that the level of intra-operator variabilityw ould be reduced if the expert raters had more practice with the tools.

6.3.3.3 Inter-operator variability

The next experiment concerns inter-rater variability. First, we discuss an issue raised by the contouring results, regarding the difference between users, in their *perception* of lesion boundaries (figure 6.24). The freehand contours overlaid in figure 6.24 reveal that, while some lesions such as (i) and (ii) are unambiguous, others such as (iii) and (iv) are perceived differently by at least one rater, who groups nearby lesions inside the same contour. This ambiguity is distinct from the inter-rater variability used

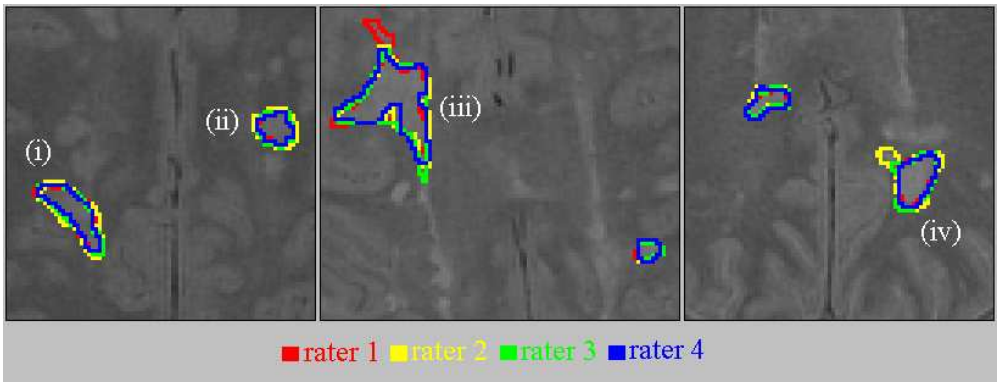


Figure 6.24: Ambiguity of MS lesion boundaries as perceived by different experts.

to assess contouring methods. We are interested in the case where lesions are perceived the same by two raters, but limitations of the method leads to discrepancies in their segmentation. We choose two unambiguous lesions (i) and (ii) to compare the inter-rater variability of each method and hypothesise that:

> $\mathcal{H}$6.3.3.3: in terms of inter-operator variability, SVM jetstreams used to segment *unambiguous* MS lesions are
>
> (a) better than the intensity-driven jetstreams, and
>
> (b) better than freehand delineation

To test hypothesis $\mathcal{H}$6.3.3.3 we measure the dissimilarity between segmentations by two users, and take the mean dissimilarity over all 6 unique pairs. Figure 6.25 compares the similarity measures for the 4 methods, shown separately for regions (i) and (ii). T-tests reveal that the region-trained SVM gives significant improvement in terms of both MMD and DSC, for ROI (i) in T2 images only. There is no significant improvement over freehand segmentation. On the whole we must not accept hypothesis $\mathcal{H}$6.3.3.3.

6.3.4 Conclusions

We have presented a generalised contouring tool that combines boundary tracking with image models based on SVM texture classification and on-line supervision. User experiments with synthetic texture images and MS lesions reveal that

- for synthetic regions with known ground-truth, jetstreams driven by texture models or intensity gradient are generally more accurate than freehand drawing,
- for MS lesions, jetstreams do not seem as accurate as freehand drawing, but the definition of accuracy is flawed in the absence of ground-truth,
- in terms of accuracy, the benefits of texture-based over intensity-based image models, as revealed by classification experiments, are largely lost when using SVMs to drive jetstreams,

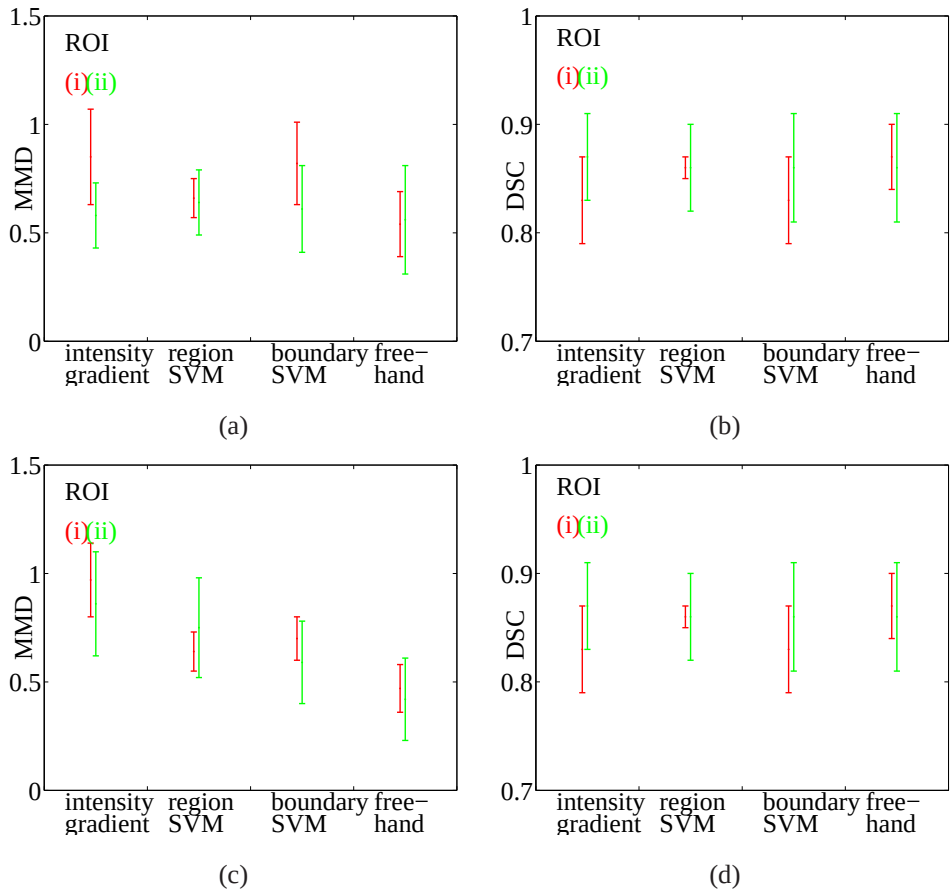


Figure 6.25: Inter-operator variability of jetstreams used for MS lesions (i) and (ii). *Top row:* for PD images by (a) mean minimum distance and (b) Dice similarity coefficient. *Bottom row:* for T2 images by (c) mean minimum distance and (d) Dice similarity coefficient.

- in terms of user demand, the benefits of texture-based over intensity-based image models, are revealed by experiments on synthetic images,

- in terms of inter- and intra-rater variability, SVM jetstreams show some improvement over free-hand segmentation in synthetic images, but statistical tests reveal no significance and the same apparent improvement is not seen for MS lesion contouring,

- the lack of improvements in segmentation variability could reflect the level of user control enabled by the jetstream framework, and the general lack of practice that users had, and

- in terms of user demand, we see significant benefits of the region-trained SVM over gradient intensity in driving boundary trackers.

6.4 Discussion and Future work

This chapter arrives at the following key conclusions:

- The strength of the SVM texture models is more apparent in classification experiments than when the models are used in supervised contouring.
- The strengths of the interactive jetstream framework are more apparent in synthetic images, where regions are larger and ground truth is known.
- The combination of texture models and jetstream interactions benefits segmentation in terms of accuracy and user demand more than variability, due to the level of user control.

The boundary tracking framework developed here enables maximal control over segmentation results, and in turn the steering mechanisms allow for idiosyncratic 'styles' of contouring. We learn from this, that while we may simultaneously maximise user control without compromising accuracy (*requirement 1*), variability may never be eradicated in a fully user-steered framework. However, this and other limitations of the tool are likely to be alleviated if users have more practice with the tool.

The methods we have developed contribute to the wider field of semiautomatic segmentation. The loop closing algorithm extends to the task of interactive post editing, where tracking between fixed points would offer a fast, accurate and repeatable method of replacing erroneous sections of a contour resulting from artefacts such as 'bleeding' in region-growing or level set frameworks.

The methods of 'no-go areas' and the ability to choose from unseen particles were deemed redundant for MS lesion segmentation, but both could be exploited differently to improve supervised segmentation. In the case of no-go areas, the method could be used to automatically exclude nearby regions that have already been segmented, in applications like MS lesion contouring, where several ROIs may exist in a given image but only one is delineated at a given time. The ability to interactively choose from multiple hypotheses in the jetstream framework may be better exploited if the choices were visible at all times rather than requiring a user to invoke their display.

The use of SVM classifiers as an image model works in the case of jetstreams, but the benefits over intensity gradient are limited due to the high levels of on-line supervision. We are motivated to extend the SVMs for use in other segmentation frameworks. The use of the region-trained SVM is particularly appealing for two reasons. First, there is evidence that this SVM improves on intensity gradient in terms of useability. Second, this SVM works well in classifying local data based on small training sets. This last observation makes the SVM suited to constraining 3-dimensional segmentation by contour propagation or making efficient use of interactions in the form of training data labelled upon initialisation.

Chapter 7

Time Series Shape Models

This chapter introduces new statistical shape models (SSMs) for variable shapes without correspondence points, which draw from the nonlinear dynamics literature and work with radial time series representations. We base the SSMs on two time series models, namely Langevin and Gaussian processes (GP) discussed in chapter 5. Both are stochastic models with deterministic components that can be exploited to characterise global dynamics. The deterministic components are the *drift* function in a Langevin model and a *kernel* function in a Gaussian process. Chapter 5 also showed that Langevin models are Markovian whereas Gaussian processes dynamics are based on continuous correlation functions over all length scales. For this reason, the Langevin models could be thought of as an extension of the Cyclic MRF in section 3.2.2 and the GP models an extension of the CAR model in section 3.2.1.

The remainder of this chapter is organised as follows. Section 7.1 describes the shape representations, introducing definitions and notations used throughout the rest of the thesis. Section 7.2 presents Langevin models for region shapes in both generalised and star-shaped cases. Section 7.3 presents generalised and star-shaped GP models. For each model, we develop machine learning procedures for training on multiple instances of a 2-dimensional shape. We also present methods of *shape scoring*, which evaluates the agreement of test shapes with a trained model. Shape scoring methods lie primarily in the field of classification and recognition, but are intended here for use in shape regularisation for segmentation. Section 7.5 tests the affinity of the models for describing MS and liver tumour ROIs and evaluates the discriminative models by using the machine learning and shape scoring procedures in classification experiments.

7.1 Shape Models and Time Series

Before introducing new shape models we must define what we mean by 'shape', as a universal definition is not available. In general, two objects have the same shape if they share certain spatial properties. In one popular definition, Kendall [273] defines these properties as

> "all the geometrical information that remains when location, scale and rotational effects are removed from an object".

According to Kendall's definition, two regions of interest in the same semantic class may not have the same 'shape'. Regions such as MS lesions or tumours can be very different because of the pathological

processes that form them. This is in contrast with anatomical regions such as vertebrae or hearts which, mutations notwithstanding, bare global similarities by genetic design. For this project we require a more general definition if we maintain that two MS lesions, for example, have the same 'shape'. We hereby define shape as

> "all the information about a region's form that is shared by all regions belonging to the same semantic class",

where the 'information about a region's form' is independent of that region's location. Note that this definition leads to a non-Euclidean description of shape, where the different 'sizes' of shapes in a class form part of the model for that class. This is applicable for pathological applications where regions of different scale are expected due to their growth. For the statistical shape models developed here, the 'information' is embedded in the choice of deterministic function and its parameters. The remainder of this section defines various components of the parametric contours and shape models central to the remainder of the thesis.

7.1.1 Radial time series

A family of parametric contours uses the radius of N successive points around a region boundary $\mathbf{r} = \{r_0, \ldots, r_{N-1}\}$, measured from a fixed location $\mathbf{x}_c = \{x_c, y_c\}$ inside the region [163, 73, 78]. Section 3.5.1 concluded that this representation benefits from knowledge of the ROI centre, and also removes the assumption of correspondence. We refer to $\mathbf{r}$ generally as a *radial time series*, where specific types differ by what 'time' represents. Two examples are boundary arc-length s (eg. [166]) and the angle θ between radial vectors (eg. [158]). Formally, we define a shape in each case by a parameter set $\mathbf{Q}$ comprising

$$\begin{aligned} \mathbf{Q}_{\text{gen}} &= \{\mathbf{r}, \mathbf{s}, \mathbf{x}_c\} = \{\{r_0, \ldots, r_{N-1}\}, \{s_0, \ldots, s_{N-1}\}, \{x_c, y_c\}\} \quad \text{(a)} \\ \mathbf{Q}_{\text{star}} &= \{\mathbf{r}, \theta, \mathbf{x}_c\} = \{\{r_0, \ldots, r_{N-1}\}, \{\theta_0, \ldots, \theta_{N-1}\}, \{x_c, y_c\}\} \quad \text{(b)}, \end{aligned} \tag{7.1}$$

where the generalised case $\mathbf{Q}_{\text{gen}}$ in 7.1(a) can represent any two-dimensional shape, while the polar representation 7.1(b) is limited to the 'star-shaped' set where all boundary points of a shape, denoted $\mathbf{Q}_{\text{star}}$, are visible from $\mathbf{x}_c$. The polar representation has the benefit of naturally representing closed contours without self-intersection.

In general terms, a time series models the discrete time evolution of a *state variable*. In our case the state variable is radius r. For the dynamical models introduced later, it is convenient to work in the state space of a *zero-mean field* as in [75, 73, 77, 225]. In a zero-mean field, the mean of a series of length N approaches zero as $N \to \infty$, although the mean of any finite series is arbitrary. We denote as $\chi(t)$, an arbitrary state variable of a zero-mean field. Figure 7.1 illustrates the relationship between zero-mean field, radial time series and shape for both star shaped and generalised parametrisations.

7.1.2 Training data

The discriminative shape models in this section can use either the generalised or star-shaped parametrisations in equation (7.1). In either case, training data is originally in the form of closed boundaries

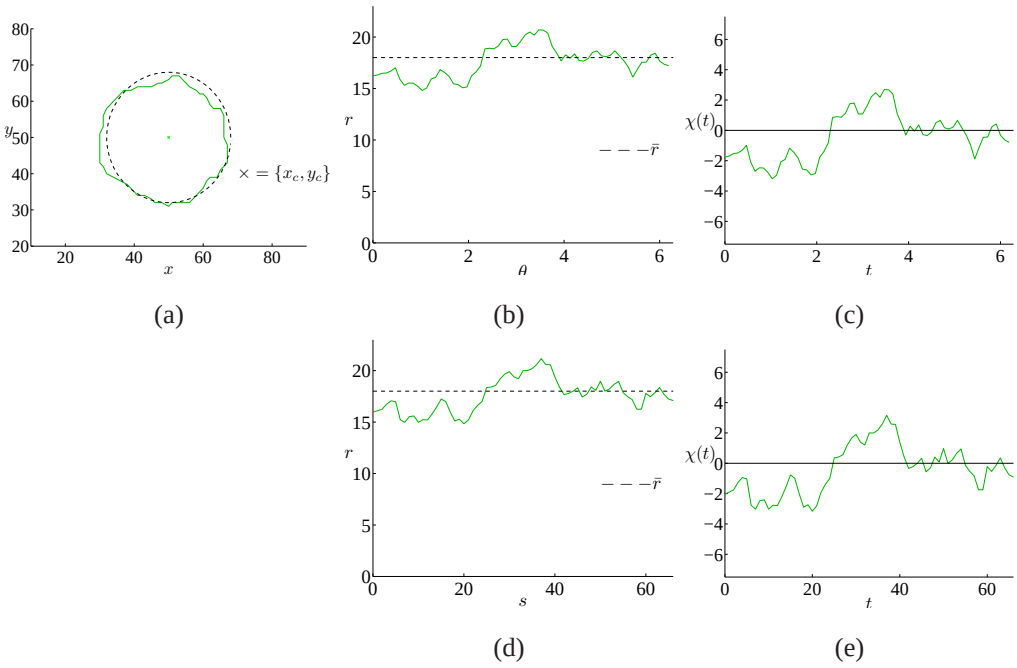


Figure 7.1: Time series shape model definitions, showing how a shape (a) relates to the star-shaped representation of (b) radial time series and (c) time series in a zero-mean field, and the generalised parametrisation of (d) radial time series and (e) time series in a zero-mean field. Dashed lines in (a), (b) and (d) show the centre of the radial state space.

expressed in $\{x, y\}$ coordinates. We pre-process this data to create radial time series with three required properties. First, training series should be of zero-mean field as described above. Second, the independent variable θ or s needs to be discretised at regular intervals. Third, in the case of GP models, the length of each series N is the same.

We first choose an internal point $\mathbf{x}_c$ to represent a regions centre. This is the only additional information we need to assign to our training contours, in contrast to the landmarks required in the point distribution model (section 3.1.1) or the skeleton in the M-reps method (section 3.1.2). Manual estimates are simple to obtain but time consuming, so we introduce automatic methods for both generalised and star-shaped training data. In the generalised case, we take the internal point having maximum closest distance to any of the boundary pixels. In the star-shaped case, we first identify the internal points from where all boundary points are visible, known as the 'kernel' of the shape. If no such points exist we omit the contour from the star-shaped set, otherwise we compute $\mathbf{x}_c$ from the centroid of these points. We refer a centre point defined by the automatic methods above as the *true* centre, as it is used when training a model.

After identifying $\mathbf{x}_c$ we record the series of radial distances r_i at each pixel around a training contour, along with the corresponding increments of the independent variable θ or s. In the star-shaped case, the angles θ_i are monotonically increasing but arbitrarily spaced, so we resample the series at regular $\Delta\theta$. For the GP models we also resample all generalised series $\{\mathbf{r}, \mathbf{s}\}$ to have common N. Next, we estimate $\bar{r}$. This radius represents the centre of the state space occupied by the radial time series,

corresponding to zero in a zero-mean field. To estimate $\bar{r}$ we take the midpoint between the minimum and maximum radius in a series. Recall that the mean of a given (finite) series does not relate to the centre of state space. Finally, we subtract $\bar{r}$ from the series, which splits the model into a zero-mean field time series and a separate scale parameter $\bar{r}$. This scale parameter $\bar{r}$ is analogous to the parameter α in the CAR model (section 3.2.1). Note that the models could be made scale invariant at this stage, either by dividing all radii by the scale parameter or by normalising radial time series to a fixed range such as $\{-1, \ldots, +1\}$, for applications where scale invariance is preferred.

7.2 Langevin Models

Section 5.1 reviewed the use of Langevin models to characterise and simulate dynamical systems including physiological processes [232] and series in the spatial domain [224]. This section formulates Langevin-type models for radial time series, with the goal of modelling fluctuating boundaries of ROIs such as tumours and lesions. The models learn higher-level information about the global statistics of shapes, which generalises despite high within-class variability.

Langevin shape models treat a radial time series as the discrete time evolution of a 1-dimensional state variable $r(s)$ or $r(\theta)$, which replace the vector $\chi(t)$ in section 5.1. For convenience we use the notation $r(t)$ to refer to both parametrisations. A Langevin model is characterised by the deterministic (drift) term and a stochastic (diffusion) term in the Langevin equation. We use the drift and diffusion functions of a Langevin equation to encode higher-level information about the global statistics of shapes with a small number of model parameters.

We start by writing equation 5.1 in terms of radial time series

$$\frac{\mathrm{d}r}{\mathrm{d}t} = A(r(t)) + B(r(t))\omega_t, \tag{7.2}$$

where the continuous variable r occupies a zero-mean field after subtracting an estimate of $\bar{r}$ and ω_t is time dependent Gaussian noise with zero mean and unit variance. The drift and diffusion terms model the 'stability' of a boundary as a function of its distance from the region centre, where the drift relates to whether the boundary shifts toward or away from the centre, and the diffusion relates to the strength of this tendency.

7.2.1 Drift and diffusion functions

The drift term $A(r(t))$ in equation 7.2 allows the local dynamical behaviour of a boundary to vary throughout the state space of 'radius'. This gives a rotation-invariant model of the global characteristics of boundary fluctuations. For the purpose of ROI shape priors, we seek drift functions that are simple, controlled by few parameters, and allow intuitive interpretation with respect to the shape itself. Where Langevin models are used for different applications in the literature, drift functions are suggested according to a combination of empirical evidence and any knowledge of an underlying physical model [234, 232, 224, 274, 236, 242]. For a stable series to remain in a zero-mean field, we require that the drift function has a global trend of $A(r) \to A_{-\infty} \geq 0$ as $r \to -\infty$ and $A(r) \to A_{+\infty} \leq 0$ as $r \to +\infty$. We refer to this as the *negative trend requirement*.

We present three simple candidates for drift and diffusion functions, chosen after empirical investigations to model tumour and lesion regions. The candidate drift ($A_{1\ldots3}$) and diffusion functions ($B_{1\ldots3}$) are given by

$$\begin{aligned} A_1(r(t),\mathbf{a}) &= -a_0 r \exp[-r^2/a_1^2] - a_2 r \qquad & B_1(r(t),\mathbf{b}) &= b_0 \\ A_2(r(t),\mathbf{a}) &= a_0 r + a_1 r^3 - a_2 r^5 & B_2(r(t),\mathbf{b}) &= b_0 + b_1(r-b_2)^2 \\ A_3(r(t),\mathbf{a}) &= a_0 - a_1 \exp[a_2 r] & B_3(r(t),\mathbf{b}) &= b_0 + b_1(r-b_2)^3. \end{aligned} \tag{7.3}$$

We ensure the drift functions $A_{1\ldots3}$ fulfil the negative trend requirement by asserting a_2 is positive in functions A_1 and A_2, and a_1 is positive in A_3.

7.2.2 Parameter estimation

Starting with a set of training shapes, the machine learning task is to estimate the scale parameter $\bar{r}$ in $\mathbf{Q}_{\text{gen}}$ or $\mathbf{Q}_{\text{star}}$ and the form and parameters of the drift and diffusion functions.

For the scale parameter we assume a normal distribution $\Pr(\bar{r}) = \mathcal{N}(\hat{\bar{r}}, \sigma_{\bar{r}}^2)$ and calculate the mean $\hat{\bar{r}}$ and standard deviation $\sigma_{\bar{r}}$ by estimating $\bar{r}$ from the radial time series of each training shape.

To estimate the form and parameters of the drift and diffusion function we devise machine learning procedures adapted from the direct estimation method of Friedrich and Peinke [229]. The adaptations enable us to learn from multiple instances of periodic series derived from a training set of shapes.

First we transform a set of M training shapes into zero-mean series $\{\mathbf{r}^0, \ldots, \mathbf{r}^m, \ldots, \mathbf{r}^{M-1}\}$ as described in section 7.1.2. Next we form a discrete estimation of the drift and diffusion functions common to the population of training shapes in the following steps:

Step 1. Divide the state space $\{r_{\text{min}}, \ldots, r_{\text{max}}\}$ of the whole training set, into bins of equal width Δr, centred on discrete values r_n.

Step 2. For the n^{th} bin, inspect the m^{th} series and note all observations $r^m(t_i)$ that fall in the range $r^m(t_i) \in r_n \pm \frac{\Delta r}{2}$.

Step 3. Starting from these observations, follow the series r_i^m along a trajectory of length Δt. Where this trajectory overshoots the end of a series, re-start from r_0 as the series is periodic.

Step 4. Repeat steps 2-3 for all M series in the training set to form a common histogram from future positions of the state variable.

Step 5. Use this histogram to approximate the transition density as $\Pr(r(t+\Delta t)|r(t) \in r_n \pm \frac{\Delta r}{2}) = \mathcal{N}(\mu_n, \sigma_n)$ specific to the n^{th} bin in state space.

Step 6. Estimate the mean μ_n and standard deviation σ_n from the normal distribution $\mathcal{N}(\mu_n, \sigma_n)$.

These steps arrive at discrete approximations of the drift and diffusion functions

$$\begin{aligned} A(r_n(t)) &= \mu_n - r_n \qquad \text{and} \\ B(r_n(t)) &= \sigma_n \qquad r_n \in \{r_{\text{min}}, \ldots, r_n, \ldots, r_{\text{max}}\}, \end{aligned} \tag{7.4}$$

specific to the chosen delay parameter Δt. Next we simultaneously seek functions $A(r(t), \mathbf{a})$ and $B(r(t), \mathbf{b})$ along with their parameters $\mathbf{a}$ and $\mathbf{b}$, which best fit the observed drift and diffusion functions. We use a Levenberg Marquardt fitting routine and start by fitting each of equations $A_{1\ldots3}$ to the extracted drift functions and each of $B_{1\ldots3}$ to the extracted diffusion functions. Then we take the combination of functions A and B whose fits give the lowest χ^2 error. For both A and B we omit the observations at the extremes of state space from the fitting procedure, as these regions are under-represented in the training data.

The procedure above involves secondary parameters that must be chosen, namely the delay parameter Δt and the width of the state-space bins Δr. We optimise for these parameters for a given application by using the χ^2 error of the fitting procedure as a measure of model affinity. We also benefit from the ability to generate synthetic data (explained in the next chapter) to gain insight into the sensitivity of parameter estimation to these values.

7.2.3 Shape scoring

Shape scoring assigns a value to a test shape, according to its agreement with a model. The same principle underlies shape classification and object recognition tasks. We suggest an intuitive, fast approach to shape scoring, for the purpose of shape regularisation in region segmentation.

Where time series models have been used for classification in the literature, the general approach is to repeat parameter estimation for both training and testing data, and score the test set based on the similarity of the two estimates. Examples for shape classification by the circular autoregressive model [159, 160] use estimated parameters as feature vectors in generic classification schemes. An example for electrocardiograph (ECG) series classification by the Langevin model [223], groups ECG series into healthy and unhealthy classes by qualitative comparison of the estimated drift functions. These approaches assume that a single test shape/series contains sufficient information for parameter estimation. In the case of the circular autoregressive model, parameters are reliably estimated from a single shape. In the example of ECG, a typical series contains tens of thousands of points, representing heart rate fluctuations over periods up to 24 hours.

These approaches to shape scoring are not appropriate for our purposes for three reasons. First, for the purpose of shape regularisation, we need to score the agreement of a single time series with a model. The parameter estimation procedure above can not reliably estimate parameters from a single radial time series, which does not contain sufficient data. Second, for shape regularisation in an interactive segmentation framework the scoring method must be fast, whereas the approaches above are likely to be slow. Third, we prefer a score that is probabilistic, in order to allow integration into a probabilistic segmentation framework as motivated in section 2.5.

We introduce a probabilistic shape scoring method based on the transition densities derived from the Fokker-Planck equation. Formally, for a contour defined by parameters $\mathbf{Q} = \{\mathbf{r}, \mathbf{x}_c, \bar{r}\}$, we seek the prior probability $\Pr(\mathbf{Q}|\mathbf{a})$. This chapter is not concerned with the position of a contour model relative to an image, so we omit the dependence on $\mathbf{x}_c$ and score the shape according to $\Pr(\mathbf{Q}|\mathbf{a}) = \Pr(\{\mathbf{r}, \bar{r}\}|\mathbf{a})$. For Langevin shape priors we recall the expression for the joint likelihood which, or the case of the shape

models, becomes

$$\Pr(\mathbf{Q}|\mathbf{a},\mathbf{b}) = \Pr(r_0)\left[\prod_{i=0}^{N-1}\Pr(r(t_i+\Delta t)|r(t_i),\mathbf{a},\mathbf{b})\right]\Pr(\bar{r}). \tag{7.5}$$

We finally define the score for a single shape, by dividing the joint likelihood by the number of points around the contour and taking the logarithm

$$\begin{aligned} S_{\text{LAN}} &= \frac{1}{N}\times\log\Pr(\mathbf{Q}|\mathbf{a},\mathbf{b}) \\ &= \frac{1}{N}\times\left(\log\Pr(r_0)+\left[\sum_{i=0}^{N-1}\log\Pr(r(t_i+\Delta t)|r(t_i),\mathbf{a},\mathbf{b})\right]+\log\Pr(\bar{r})\right), \end{aligned} \tag{7.6}$$

where $\Pr(\bar{r}) = \mathcal{N}(\hat{\bar{r}},\sigma_{\bar{r}}^2)$ is the distribution over scale parameters and we set $\Pr(r_0) = \Pr(\bar{r})$ without loss of generality because the first point in a radial time series can be chosen at any point on the boundary. The conditional probabilities $\Pr(r_i|r_{i-1})$ are normally distributed with means and variances given by

$$\Pr(r(t_i)|r(t_{i-1})) = \mathcal{N}(r(t_{i-1})) - A(r(t_{i-1}),\mathbf{a}),B(r(t_{i-1}),\mathbf{b})), \tag{7.7}$$

where A and B are evaluated using the learned model parameters $\mathbf{a}$ and $\mathbf{b}$.

7.3 Gaussian Process Models

A Gaussian process (GP) is characterised by a discrete mean function and a kernel function that defines a covariance matrix. Section 5.2 reviewed GP methods, noting that the 1-dimensional case is an example of nonlinear time series analysis. This section formulates GP models for radial time series, with the goal of learning higher-level information about the global statistics of boundary dynamics. In this section we introduce GP SSMs with an example kernel function and constant mean function, and discuss extensions to other kernels and non-constant mean function as the basis of future work.

Gaussian process shape models treat a radial time series as a random vector of radii $\mathbf{r}$ at discrete inputs $\mathbf{s}$ or θ. As above we use the general notation $r(t)$, which replaces $\chi(t)$ in section 5.2. We propose that, by modelling a radial time series as a Gaussian process, the kernel function encodes higher-level shape information with a small number of model parameters.

We start by rewriting equation 5.11 to represent a radial time series as a N-dimensional random vector of outputs $\mathbf{r} = \{r_0, r_1, \ldots, r_{N-1}\}$ corresponding to inputs $\mathbf{t} = \{t_0, t_1, \ldots, t_{N-1}\}$. A given series $\mathbf{r}$ has an associated probability $\Pr(\mathbf{r}|\mu,\mathbf{\Sigma}(\mathbf{r},\mathbf{r}))$, which follows the N-dimensional multivariate normal distribution

$$\begin{aligned} \Pr(\mathbf{r}|\mathbf{\Sigma}(\mathbf{r},\mathbf{r})) &= \mathcal{N}_N(\mu,\mathbf{\Sigma}(\mathbf{r},\mathbf{r})) \\ &= \frac{1}{2\pi^{\frac{N}{2}}|\mathbf{\Sigma}(\mathbf{r},\mathbf{r})|^{\frac{N}{2}}}\exp[-\frac{1}{2}(\mathbf{r}-\mu)^T\mathbf{\Sigma}^{-1}(\mathbf{r},\mathbf{r})(\mathbf{r}-\mu)] \end{aligned} \tag{7.8}$$

where μ is a discrete mean function given by the vector of expectation values $E(r_i)$ and $\mathbf{\Sigma}(\mathbf{r},\mathbf{r})$ is the $N\times N$ covariance matrix. Elements of $\mathbf{\Sigma}$ model the covariance between pairs of outputs $\{r_i, r_j\}$ as a function of the corresponding inputs $\{t_i, t_j\}$. The covariance matrix is computed using the kernel function

$$\Sigma(r_i,r_j) = \varepsilon(t_i,t_j,\mathbf{a}), \qquad i,j\in\{0,\ldots,N-1\}, \tag{7.9}$$

where $\mathbf{a}$ is a vector of parameters of the kernel function. The covariance matrix becomes

$$\mathbf{\Sigma}(\mathbf{r},\mathbf{r}) = \begin{pmatrix} \varepsilon_{0,0}(t_0,t_0,\mathbf{a}) & \varepsilon_{0,1}(t_0,t_1,\mathbf{a}) & \dots & \varepsilon_{0,N-1}(t_0,t_{N-1},\mathbf{a}) \\ \varepsilon_{1,0}(t_1,t_0,\mathbf{a}) & \varepsilon_{1,1}(t_1,t_1,\mathbf{a}) & \dots & \varepsilon_{1,N-1}(t_1,t_{N-1},\mathbf{a}) \\ \vdots & \vdots & \ddots & \vdots \\ \varepsilon_{N-1,0}(t_{N-1},t_0,\mathbf{a}) & \varepsilon_{N-1,1}(t_{N-1},t_1,\mathbf{a}) & \dots & \varepsilon_{N-1,N-1}(t_{N-1},t_{N-1},\mathbf{a}) \end{pmatrix}. \quad (7.10)$$

7.3.1 Kernel and mean functions

The kernel function (equation 7.9) provides the deterministic part of the GP model, which describes how the correlation between two boundary points varies with their separation. We choose a stationary kernel $\varepsilon(t_i,t_j) = \varepsilon(t_i - t_j,\mathbf{a})$. In this way the kernel functions are both rotation invariant and easily interpreted, modelling correlation as a function of *length-scale* $t_i - t_j$. For the case of radial time series we seek a kernel function that is simple, and periodic to ensure correlation between the points at the beginning and end of a series. We use a function based on the periodic kernel in [275], given by

$$\varepsilon(t_i - t_j,\mathbf{a}) = \exp\left[-a\sin^2\left(\frac{t_j - t_i}{2}\right)\right], \quad (7.11)$$

which has a single free parameter $\mathbf{a} = \{a\}$ governing the length-scale of correlation.

For convenience, and to retain rotation invariance, we use a constant mean function. In the zero-mean field this corresponds to the vector of zeros $\mu = \mathbf{0}$. Depending on the application, novel mean functions could be derived from training data or run-time interactions in a segmentation framework.

7.3.2 Parameter estimation

Starting with a set of training series, the machine learning task is to estimate the discrete mean function and the form and parameters of the kernel function. In this section we assume constant expectation values $E(r_i) = 0 \forall i$ such that the mean function μ is a vector of zeros. We also assume that the kernel function in equation 7.11 is general enough to describe any given training set. The remaining task is to estimate the mean and variance of $\mathcal{N}(\hat{\bar{r}},\sigma^2_{\bar{r}})$, and the parameters $\mathbf{a}$ of the kernel function. As before we estimate $\bar{r}$ for a single contour as above and take the mean $\hat{\bar{r}}$ and standard deviation $\sigma_{\bar{r}}$ over the training set. To estimate the kernel parameters we use Markov Chain Monte Carlo (MCMC) methods following the work of [248].

We choose the MCMC method for its ability to avoid local minima and, moreover, to handle *multitask learning* whereby, as in our case, the training data naturally come as multiple independent series. Starting with M training shapes $\mathbf{r}^m$, where $m = 0,\dots,M-1$, the algorithm seeks the parameters $\mathbf{a}$ that maximise the joint probability density function $\Pr(\mathbf{r}^{0,\dots,M-1}|\mathbf{a})$ given by

$$\begin{aligned} \Pr(\mathbf{r}^{0,\dots,M-1}|\mathbf{a}) &= \prod_{m=0}^{M-1} \Pr(\mathbf{r}^m|\mathbf{a}) \\ &= \prod_{m=0}^{M-1} \frac{1}{2\pi^{\frac{N}{2}}|\mathbf{\Sigma}(\mathbf{a})|^{\frac{N}{2}}} \exp[-\frac{1}{2}(\mathbf{r}^m - \mu)^T\mathbf{\Sigma}^{-1}(\mathbf{a})(\mathbf{r}^m - \mu)], \end{aligned} \quad (7.12)$$

where $\mathbf{\Sigma}(\mathbf{a})$ denotes that the free parameters $\mathbf{a}$ are the only unknowns in the covariance matrix. For convenience and to avoid numerical issues arising from near singular covariance matrices [166], we

maximise the log of the joint PDF given by

$$\begin{aligned} L &= \log \Pr(\mathbf{r}^{0,\ldots,M-1}|\mathbf{a}) \\ &= -\frac{MN}{2}\log(2\pi) - \frac{M}{2}\log(|\mathbf{\Sigma}(\mathbf{a})|) - \frac{1}{2}\sum_{m=0}^{M-1}\big((\mathbf{r}^m - \mu)^T\mathbf{\Sigma}^{-1}(\mathbf{a})(\mathbf{r}^m - \mu)\big). \end{aligned} \tag{7.13}$$

The MCMC algorithm seeks the posterior distribution over parameters $\pi(\mathbf{a})$. We use Gibbs sampling to repeatedly draw samples from a proposal distribution. For the case of equation 7.11 we use the 1-dimensional distribution centred on the current estimate a_i, i.e. $\mathcal{N}(a_i, \sigma_a)$, where variance σ_a is chosen empirically. At the $i+1^{\text{th}}$ iteration, the parameter a_{i+1} drawn from the proposal distribution replaces the 'current' parameter estimate a_i with probability given by the likelihood ratio $\frac{L(a_{i+1})}{L(a_i)}$ until, after a 'burn-in' period, the Markov Chain iteratively samples from the stable distribution $\pi(\mathbf{a})$. If we assume $\pi(\mathbf{a})$ to be Gaussian then the maximum *a-posteriori* probability (MAP) solution is given by the mean

$$\frac{1}{k}\sum_{i=\mathcal{B}}^{\mathcal{B}+k} a_i, \tag{7.14}$$

over a large number k of samples, where $\mathcal{B}$ is the number of iterations in the burn-in period.

7.3.3 Shape scoring

Proceeding as for the Langevin model, we seek to score a test shape according to its agreement with the model. As before, we evaluate $\Pr(\mathbf{Q}) = \Pr(\{\mathbf{r}, \bar{r}\})$, i.e. independent of the centre point $\mathbf{x}_c$. We start with the log probability density function for a single series, given the learned parameters $\mathbf{a}$,

$$\log(\Pr(\mathbf{r}|\mathbf{a})) = \frac{N}{2}\log(2\pi) - \frac{1}{2}\log(|\mathbf{\Sigma}(\mathbf{a})|) - \frac{1}{2}\big((\mathbf{r}-\mu)^T\mathbf{\Sigma}^{-1}(\mathbf{a})(\mathbf{r}-\mu)\big). \tag{7.15}$$

The full equation for shape scoring under the GP model includes the scale parameter $\Pr(\bar{r}) = \mathcal{N}(\hat{\bar{r}}, \sigma^2_{\bar{r}})$, giving

$$\begin{aligned} S_{\text{GP}} &= \log \Pr(\mathbf{Q}|\mathbf{a}) \\ &= \log(\Pr(\mathbf{r}|\mathbf{a})) + \log \Pr(\bar{r}) \\ &= \frac{N}{2}\log(2\pi) - \frac{1}{2}\log(|\mathbf{\Sigma}(\mathbf{a})|) - \frac{1}{2}\big((\mathbf{r}-\mu)^T\mathbf{\Sigma}^{-1}(\mathbf{a})(\mathbf{r}-\mu)\big) + \log \Pr(\bar{r}). \end{aligned} \tag{7.16}$$

7.4 Data and Performance Evaluation

We have 276 liver tumour contours from [29], of which 241 are star-shaped. We also have 3086 MS lesion contours from the datasets in the previous chapter. To reduce this large training set we discard the smallest MS lesions, made up of 15 pixels or less. The remaining ground truth are 1608 MS lesion contours, of which 1307 are star-shaped. Figure 7.2 (a) and (b) show examples of liver tumour and MS lesion contours respectively.

7.4.1 Figures of merit

The performance of a discriminative model is related to its specificity and sensitivity. Specificity tells us how consistently a model describes a specific region type. In the case of Langevin models, the specificity of candidate functions in equation (7.3) predicts their relative discriminator power. For this purpose we infer Langevin model specificity using the χ^2 error returned by Levenberg-Marquardt fitting.

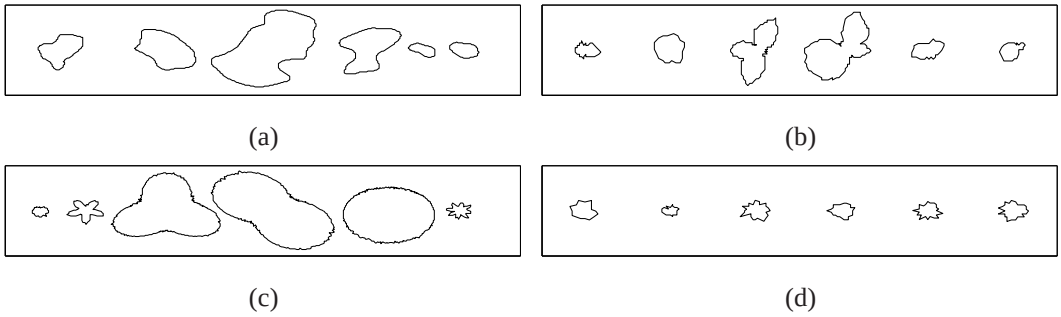


Figure 7.2: *Top row*: examples of positive class ground truth shapes from (a) liver tumour and (b) MS lesion sets. *Bottom row*: examples of negative class sets defined in section 7.4.2 and used in the experiment of section 7.5.2, comprising (c) noisy sinusoids with radial range matching the liver tumour sets and (d) noisy circles with radial range matching the MS lesion set

For both Langevin and GP models, the specificity and sensitivity tell us how well a model can distinguish shapes that belong to the model from those that do not. The shape models should score test shapes with higher values S_{LAN} or S_{GP}, if they belong to the same semantic class as the training data. We evaluate a model's ability to distinguish between shapes by thresholding the scores assigned to positive and negative test shapes, and creating ROC curves. As we saw in section 3.4, the choice of negative class is somewhat arbitrary when evaluating shape models. We use synthetic shapes created to satisfy two requirements. First, a negative class must have similar radial statistics to the positive class, so that remaining differences between positive and negative class are subtle. Second, a shape must be of the type encountered during the evolution of a stochastic active contour. This last criterion allows performance evaluation to infer the value of the discriminative models as shape regularisers in a stochastic segmentation framework.

7.4.2 Negative classes

We create two synthetic negative classes that satisfy the requirements above. We refer to the first as *noisy circles*, which we generate by drawing random values r_i from a normal distribution. Each series has the same length N and first/second order statistics as one in the positive test data. Examples of noisy circles are shown in figure 7.2 (c). The second synthetic class are *noisy sinusoids* defined by $r_i = \omega \times \gamma \sin(p\ 2\pi\ i/N)$, with noise ω is drawn from $\mathcal{N}(0, 1)$ and the number of periods p is drawn from a uniform distribution between 1 and 10. The amplitude γ and length N of each series match the radial range and length N of one in the positive class. Examples of noisy sinusoids are shown in figure 7.2 (d).

7.4.3 Comparison methods

Finally, we choose comparison methods to investigate the discrimination characteristics of the SSMs. As the models are new, we are view their evaluation as a 'proof of concept', more so than a direct comparison of the state of the art. Moreover, the state of the art is not obvious, when considering the balance between maximising the shape information sought and minimising the level of shape similarity

assumed. We seek comparison methods that

- gain information about a shape without assuming correspondence, where
- the information is interpretable and
- can be adapted to give a probabilistic score based on a training population

We choose two shape descriptors that satisfy these requirements. The first descriptor is based on the sum of a local smoothness measure $\zeta = 1/N \times \sum_{i=0}^{N-1} \cos(\varphi)$ where φ is the angle between successive steps from one boundary point to the next. We calculate the mean $\hat{\zeta}$ and standard deviation σ_ζ of all training contours assuming a normal distribution $S_\zeta = \mathcal{N}(\hat{\zeta}, \sigma_\zeta)$. A test contour with smoothness ζ' is scored using the normalised log probability given by

$$S_\zeta = \frac{1}{N}\big[\frac{1}{2}\log(2\pi) - \frac{1}{2}\log(\sigma_\zeta) - (\frac{\zeta' - \hat{\zeta}}{\sigma_\zeta})^2\big] + \log \Pr(\bar{r}). \tag{7.17}$$

The second descriptor is based on the 1-dimensional Fourier decomposition of a radial time series. We estimate the first K coefficients β^s of sine and β^c of cosine terms in the approximation $\mathbf{r} \approx \sum_{k=0}^{K-1} \beta_k^c \cos(kt) + \beta_k^s \sin(kt)$. For each training contour we form the $2K$-dimensional vector $\beta = \{\beta_0^s, \ldots, \beta_{K-1}^s, \beta_0^c, \ldots, \beta_{K-1}^c\}$ and calculate the mean vector $\hat{\beta}$ and covariance matrix $\Sigma(\beta, \beta)$ assuming a multivariate normal distribution $\mathcal{N}_{2K}(\hat{\beta}, \Sigma(\beta, \beta))$. A test contour with Fourier coefficients β' is scored using the normalised log probability given by

$$S_\mathcal{F} = \frac{1}{N}\big[\frac{1}{2}\log(2\pi) - \frac{1}{2}\log(|\mathbf{\Sigma}(\beta, \beta)|) - \frac{1}{2}\big((\beta' - \hat{\beta})^T \mathbf{\Sigma}^{-1}(\beta, \beta)(\beta' - \hat{\beta})\big)\big] + \log \Pr(\bar{r}). \tag{7.18}$$

7.5 Experiments

This section tests the discriminative power of the dynamical SSMs. We address two hypotheses concerning the efficacy of the Langevin models for the chosen region types (7.5.1) and the sensitivity of both Langevin and GP models (7.5.2).

7.5.1 Langevin model selection for medical contours

In the case of Langevin models we have suggested more than one function to describe the observed dynamics of a training set. The next experiments investigate different drift and diffusion functions for a given region type, in order to test that a Langevin model adequately describes the region type, and choose the best functions.

We hypothesise that:

> $\mathcal{H}$7.5.1: Langevin models capture global shape information specific to a population of region boundaries.

We address hypothesis $\mathcal{H}$7.5.1 in two ways. First, we assess the models qualitatively by looking for structure in the discrete drift and diffusion functions extracted from training data. We use the steps in section 7.2.2 to estimate drift and diffusion functions by equation (7.4). Figure 7.3 shows the results. By visual inspection, drift and diffusion functions have some structure for both region types modelled

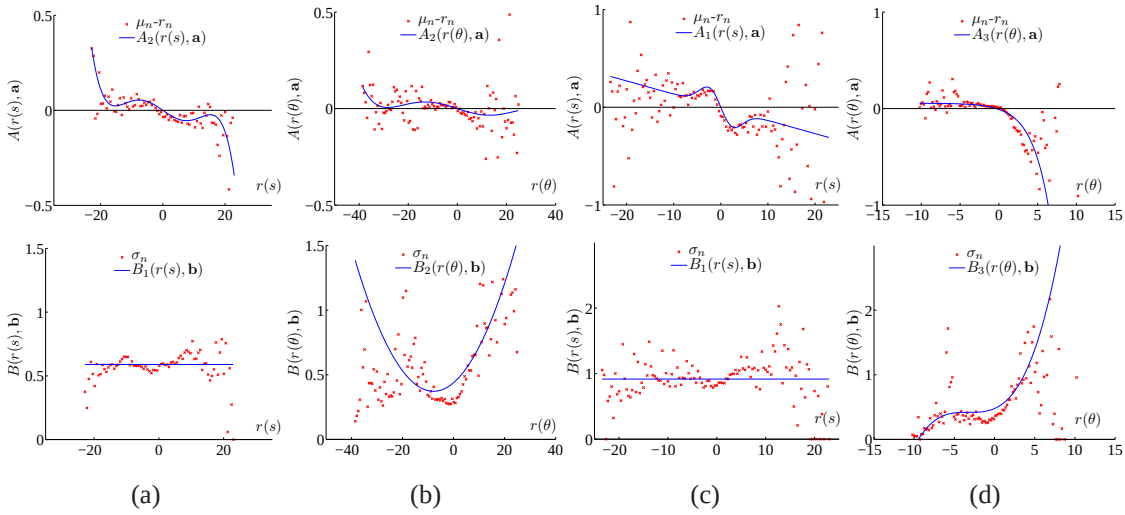


Figure 7.3: Extracted drift (top row) and diffusion (bottom row) functions for Langevin models trained on (a)/(b) liver tumours and (c)/(d) MS lesions using the generalised/star-shaped parametrisations respectively.

with both generalised and star-shaped parametrisations. In all cases, however, the structure is degraded towards the extremes of state space where training data is sparse.

Second, we look for evidence that different region types are best described by different (combinations of) drift and diffusion functions. Table 7.1 shows the χ^2 error when each candidate function is used for each region-type and both generalised ('gen.') and star-shaperd parametrisations.

Region type	Drift	χ^2 error		Diffn.	χ^2 error	
		gen.	star		gen.	star
liver tumour	A_1	2.55×10^7	1.20×10^8	B_1	$\mathbf{6.25 \times 10^7}$	3.49×10^8
	A_2	$\mathbf{2.21 \times 10^7}$	$\mathbf{1.19 \times 10^8}$	B_2	5.82×10^7	$\mathbf{2.54 \times 10^8}$
	A_3	2.81×10^7	1.21×10^8	B_3	6.00×10^7	2.71×10^8
MS lesion	A_1	$\mathbf{3.63 \times 10^8}$	4.91×10^8	B_1	$\mathbf{4.25 \times 10^8}$	2.89×10^9
	A_2	3.92×10^8	4.70×10^8	B_2	3.68×10^8	1.39×10^9
	A_3	4.12×10^8	$\mathbf{2.29 \times 10^8}$	B_3	3.69×10^8	$\mathbf{1.33 \times 10^9}$

Table 7.1: χ^2 errors when fitting functions to discrete estimates of Langevin drift and diffusion.

Solid lines in figure 7.3 show the results of fitting chosen functions (bold in table 7.1) from the candidate set 7.3. The chosen functions are those having the lowest χ^2 error, in all cases except for the diffusion functions in the generalised models (bottom row, (a) and (c)). In these cases the constant and quadratic functions B_1 and B_2 give similar χ^2 error and we favour B_1 for its simplicity.

We accept hypothesis $\mathcal{H}$7.5.1 because, for both generalised and star-shaped models, we see structure in the estimated functions, and these are different for the two region types, suggesting that the training contours have distinct global properties that the Langevin models can capture.

7.5.2 Discrimination capability for medical contours

This section uses the discriminative SSMs in classification experiments. First, we hypothesise that:

> $\mathcal{H}$7.5.2: Langevin and GP models are sensitive enough to discern tumour and lesion shapes from synthetic shapes of equivalent radial scale and variance,

We test hypothesis $\mathcal{H}$7.5.2 using ROC analysis. We divide the data into training and testing sets of approximately the same size, where contours in each set originate from a subset of the MRI or CT volumes. Next we train Langevin and GP models and use equations (7.6) and (7.16) to score the testing set along with the same number of negative-class contours, taken from the synthetic sets of noisy circles and sinusoids. We threshold the scores at 500 increments and calculate the true- and false-positive fractions that form a ROC curve. The area under the curve (AUC) provides a measure of classification accuracy between 0 and 1. The central columns of table 7.2 show the results for Langevin and GP models, used to classify liver tumour and MS lesion shapes with both generalised ($r(s)$) and star-shaped ($r(\theta)$) contour parametrisations.

Positive class	Negative class	Langevin (S_{LAN})		Gauss. Proc. (S_{GP})		Smooth (S_ζ)	Fourier ($S_\mathcal{F}$)	
		$r(s)$	$r(\theta)$	$r(s)$	$r(\theta)$		$K=3$	$K=10$
Liver tumour	circular	0.989	0.999	0.861	0.978	0.961	0.644	0.796
	sinusoid	0.919	0.930	0.813	0.943	0.621	0.598	0.685
MS lesion	circular	0.030	0.532	0.837	0.804	0.810	0.698	0.678
	sinusoid	0.799	0.916	0.581	0.813	0.723	0.753	0.692

Table 7.2: Classification results for the SSMs and simple shape descriptors, used to distinguish liver tumour and MS lesion shapes from synthetic negative classes.

In general, the models of liver tumour shapes perform well, with AUC above 0.9 in all cases except the GP model with the generalised contour parametrisation. The star-shaped Langevin model also discriminates between lesions and noisy sinusoids with AUC $>$ 0.9, while all other SSMs struggle to distinguish MS lesion shapes from either negative class. A striking example is discriminating MS lesions from noisy circles, which causes the Langevin model to fail and, in the generalised model, consistently misclassify (AUC$<$0.5). We revisit this observation in section 7.6. On the whole we accept hypothesis $\mathcal{H}$7.5.2 for the star-shaped models, observing that both Langevin and GP models discriminate liver tumours better than MS lesions.

Next, we hypothesise that:

> $\mathcal{H}$7.5.3: Langevin and GP models capture more information than
> (a) a linear combination of local smoothness, or
> (b) global information regarding frequency statistics

We repeat ROC analysis using the smoothness and Fourier descriptors described above, creating ROC curves by thresholding S_ζ and $S_\mathcal{F}$. In the case of Fourier descriptors we repeat for $K = 3$ and

$K = 10$ in equation 7.18, to truncate the Fourier descriptors at lower and higher frequencies. Results are given in the right hand side of table 7.2.

In most cases the dynamical SSMs have higher classification accuracy than the smoothness and Fourier descriptors. Langevin models discriminate liver tumours from noisy circles better than from noisy sinusoids, whereas this ranking is reversed for the case of MS lesions. This confirms that the two medical region types differ in terms of the dynamics captured by Langevin models. We accept hypothesis $\mathcal{H}$7.5.3.

Finally, we investigate the relative efficacy of Langevin and GP models for describing ore regions of interest, recalling that the former assumes Markovian dynamics and the latter does not. For this we make the null hypothesis

$\mathcal{H}$7.5.4: Langevin and GP models have the same discriminatory power.

Comparing the Langevin and GP columns in table 7.2 leads to the following observations:

(i) Langevin models out perform GP models in discriminating liver tumours from noisy circles and MS lesions from noisy sinusoids.

(ii) GP models out perform Langevin models in discriminating MS lesions from noisy circles.

Taken together, observations (i) and (ii) indicate that tumour and lesion boundaries both fluctuate with Markovian dynamics, but this behaviour alone does not discern MS lesions from noisy circles. However, we can not make a general conclusion about the efficacy of a Markovian model because the Langevin models use deterministic functions chosen for the respective lesion types from the set in equation 7.3, whereas we only try a single kernel function in the GP model.

7.6 Conclusions and Future Work

This chapter presented statistical shape models that combine nonlinear time series analysis with radial time series contour parametrisations. Model selection and classification experiments reveal that

- Langevin models capture global information that differs between region types, and
- Langevin and GP models distinguish tumours and lesions from synthetic shapes with similar radial statistics and range.

These findings show that the SSMs capture global information about region boundaries, without assuming correspondence points or other high-level shape similarity between training examples.

Comparisons with two simple shape descriptors reveal that

- Langevin and GP models capture global information that is separate from the smoothness or frequency of boundary fluctuations
- Langevin and GP models generally perform better than simple smoothness and Fourier descriptors, and

- star-shaped models generally out-perform those using generalised contour parametrisation.

These findings show that the SSMs capture more higher-level, global shape information than integrating local smoothness around a contour or analysing the frequency spectrum of boundary fluctuations, and that their success is helped by, but not limited to, data belonging to the star-shaped set.

The choice of negative class for use in binary classification is somewhat arbitrary and this could explain a negative result (AUC <0.5) when classifying MS lesions using the Langevin model with generalised contour parametrisation.

For a noisy circle, the transition density $\Pr(r_{i+1}|r_i)$ is equivalent to the driving noise used in creating the synthetic data, i.e. a stationary distribution with zero mean and standard deviation chosen to match the lesion training set. This corresponds to a linear drift with single stable point (negative zero crossing) at the centre of the zero-mean field. The central region of figure 7.3 (c) shows similar behaviour, and it is in this region that training data is most closely aligned with the function A_1. Misclassification is caused by the synthetic negative class predominantly occupying this region of state space. In other words, real MS lesion data with boundary dynamics that are indistinguishable from noisy circles, make up a large part of the training data and the positive testing data, but *all* of the synthetic data.

The discriminative models are expected to benefit segmentation frameworks by shape regularisation, and the investigations above were designed with this in mind. Shape classification also benefits medical applications outside the field of segmentation. A recent example is given in [10] where shape models discriminate healthy from Alzheimers patients based on the shape of brain ventricles. This use of binary classification motivates future work with the new SSMs, starting with a specific role in MS lesion imaging. It has been suggested [276, 15] that multiple sclerosis gives rise to four different 'types' of lesion. So called 'Lassmann patterns' are thought to correlate with differences in disease prognosis, with implications for the treatment of the disease. Investigations currently rely on histological studies but, if training data became available, we are motivated to develop machine learning approaches to non-invasive classification of lesion types. We propose to use the shape models introduced here, perhaps in combination with the texture models in the previous chapter.

In the case of GP models, future work could incorporate different mean functions. We chose a constant mean function above for rotation invariance, as this corresponds to a circular 'mean shape'. The GP model readily extends to a non-circular mean shape by novel mean functions μ. In an interactive segmentation framework the mean shape could be given by a 'rough outline' drawn manually on an image by the user. This is a similar idea to the use of the 'template' with the 1D-CMRF in [77].

Finally we reiterate the value of the discriminative models outside classification or segmentation applications, as methods of tomographic reconstruction and image registration can also make use of shape regularisation. In the example of registration, the presence of tumours in target and source images poses a particular problem as they are likely to have changed between the times of acquiring two images. The new SSMs are expected to benefit this task, requiring only that (i) the centre point of a tumour can be estimated in the source and target image and (ii) the training data represent the variations in tumour shape due to changes over time.

Chapter 8

Segmentation Frameworks and Generative Models

This chapter exploits the time series models in the previous chapter for segmentation. First we use the discriminative models as shape regularisers in the optimisation scheme of a simple active contour model. Then we develop generative models to form the basis of novel probabilistic segmentation algorithms. We constrain the generative models to incorporate observations from image and interactions. We present the segmentation frameworks for star-shaped regions using the polar parametrisation in equation 7.1 (b) and discuss extensions for non star-shaped models.

The rest of this chapter is organised as follows. First, section 8.1 introduces appropriate observation models derived from images. Section 8.2 describes a deformable contour model exploiting the SSMs as shape regularisers. Section 8.3 presents a generalised framework for interactive segmentation using generative shape models, which combines generative SSMs with probabilistic observations and efficient interactions. We then present the specific methods of using generative Langevin and GP models in such a framework, in sections 8.4 and 8.5 respectively. Section 8.6 describes choices regarding data and performance metrics, used to evaluate the strength of the shape priors in a segmentation framework. Experiments in section 8.7 test the value of shape priors in the various segmentation frameworks. Section 8.8 discusses the findings and draws conclusions.

8.1 Image Models and Time Series

The last chapter dealt with the prior models of shape alone, independent of position in the image, allowing us to remove $\mathbf{x}_c$ from the shape model $\mathbf{Q}_{\text{star}} = \{\mathbf{r}, \bar{r}\}$. For segmentation, we re-introduce the centre point $\mathbf{x}_c$ and introduce information from the image data $\mathcal{D}$ local to the region centre, giving the expression for the posterior

$$\Pr(\mathbf{Q}|\mathcal{D}) = \Pr(\{\mathbf{r}, \bar{r}, \mathbf{x}_c\}|\mathcal{D}). \tag{8.1}$$

This section describes ways to model the image data $\mathcal{D}$ that are consistent with the polar parametrisation (8.1.1). We also model the uncertainty of centre-point location in section 8.1.2, and introduce the full Bayesian framework in section 8.1.3.

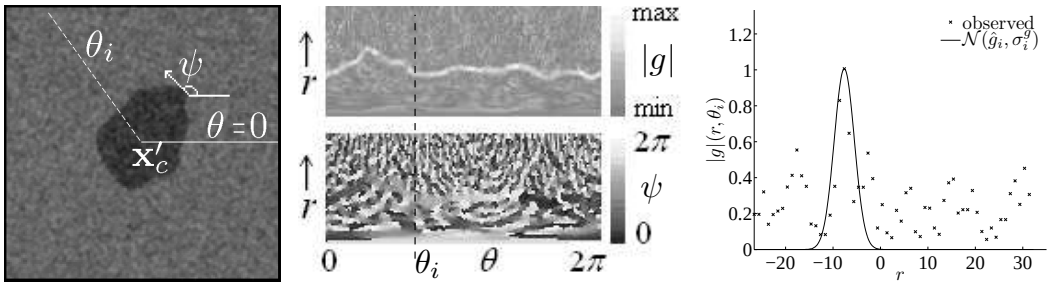
8.1.1 Data likelihood

For star-shaped regions we define an image observation model in polar coordinates. A similar method in [75] models the intensity changes at the region boundary along each radial vector as an ideal step function with Gaussian noise. For use with the GP and Langevin segmentation algorithms, we require an observation model that

- estimates the probability that the boundary intersects a radial vector at radius r,
- is independent of the choice of boundary measure, extending to texture classification if necessary
- provides observations in a form which the time series models readily incorporate, and
- is readily complemented by information from user interactions.

We introduce a *radial profile model* where the boundary measure is a function of radius. In this chapter we base the boundary measure on the gradient, denoted g, but this could be replaced by the results of tissue classification such as the boundary measure g_{d_r} or d_b used in chapter 6.

The model is based on both the magnitude $|g|$ and direction ψ of the image gradient as shown in figure 8.1. We define an estimate $\mathbf{x}'_c$ of the region centre by a pixel selected manually by the user of



(a) (b) (c)

Figure 8.1: Observation model from an example synthetic image. (a) Synthetic region with boundary given by a liver tumour contour, showing an estimate of the centre $\mathbf{x}'_c$, local boundary direction ψ and radial vector at arbitrary angle θ_i. (b) Greyscale representations of the magnitude $|g|$ (top) and direction ψ (bottom) of image gradient sampled along radial vectors, with angle θ_i marked. (c) Radial profile of gradient magnitude corresponding to angle θ_i, with Gaussian fit after translating into the zero-mean field and re-scaling to the range $\{0 \dots 1\}$.

an interactive tool. After obtaining $\mathbf{x}'_c$ we sample $|g|$ and ψ along the each radial vector θ_i. In the case of the gradient magnitude we rescale values along each profile to the range $0 \dots 1$ and fit a Gaussian function with mean $\hat{g}_i$ at the first peak of $|g|$ and standard deviation σ_i^g given by the full width at half maximum. Next we take an estimate $\bar{r}'$ of the scale parameter from the mid-point of all the profile means $\hat{g}_i, i = 0, \dots, N_{\text{obs}} - 1$ where N_{obs} is the number of observation angles. Figure 8.1 (c) shows an example radial profile of $|g|$ after translating into the zero-mean field by subtracting the estimate $\bar{r}'$. Finally, we form the likelihood ratio $p_i^{\text{on}}/p_i^{\text{off}}$, where p^{on} and p^{off} represent the probabilities that the

local section of a generated shape corresponding to $\{r_i, \theta_i\}$ is on or off the region boundary, given by

$$\begin{aligned} p_i^{\text{on}}(r) &= \exp\big[-(\psi_i(r) - \phi_i(r))^2\big], \text{and} \\ p_i^{\text{off}}(r) &= 1 - \exp\big[-(\frac{r - \hat{g}_i}{\sigma_i^g})^2\big], \end{aligned} \tag{8.2}$$

and ϕ_i is the angle with respect to the horizontal, made by the contour section from point $\{r_{i-1}, \theta_{i-1}\}$ to $\{r_i, \theta_i\}$. This definition of likelihood ratio is inspired by the joint use of gradient direction and magnitude in the jetstream algorithm of [43]. Repeating for all $i \in \{1, \ldots, N^* - 1\}$ where N^* is the number of observation angles, results in the observation model $\mathcal{D} = \{\hat{\mathbf{g}}, \sigma^g\}$.

8.1.2 Modelling centre-point uncertainty

When using the shape models in segmentation, an initial user interaction provides an estimate of the centre of an unseen ROI, denoted $\mathbf{x}'_c$. This initialisation is not precise, as a region's 'centre' does not correspond to a visual cue. For consistency, the chosen centre $\mathbf{x}'_c$ should be the same as that which would have been used, if the ROI were part of the training data, denoted $\mathbf{x}_c$. Recall from section 7.1.2, that for star a shaped regions, $\mathbf{x}_c$ is the centroid of the region's kernel. In practice, we must assume that the user-initialisation is close to the true centre $\mathbf{x}'_c \approx \mathbf{x}_c$, and incorporate uncertainty into the segmentation algorithm. The task of incorporating centre-point uncertainty is twofold. First, we seek a statistical model of the discrepancy between $\mathbf{x}'_c$ and $\mathbf{x}_c$. Second, we need to understand how this discrepancy affects a time series model in order to incorporate the effect.

For rotation invariance, we model centre-point uncertainty as an anisotropic Gaussian distribution

$$\Pr(\mathbf{x}_c|\mathbf{x}'_c) = \mathcal{N}_2(\mathbf{x}'_c, \sigma_c^2\mathbf{I}), \tag{8.3}$$

where σ_c is a common variance in x and y and $\mathbf{I}$ is the 2×2 identity matrix. We estimate σ_c using interactive experiments. We present medical images, with a region clearly delineated by its 'ground truth' contour to a volunteer who selects the pixel that they consider to be the 'centre' of the ROI. We denote an estimated centre point by $\mathbf{x}'_c$. The software then calculates the translations $\Delta x = x_c - x'_c$ and $\Delta y = y_c - y'_c$. We use these to calculate an absolute value $\Delta_c = \frac{\sqrt{\Delta x^2 + \Delta y^2}}{\bar{r}}$, divided by the scale of the corresponding region. This gives a normalised measure of the absolute 'error' the manual estimate. We repeat for 30 regions, and repeat this sequence so that the volunteer estimates the centre of each region twice, then calculate the mean value $\hat{\Delta}_c$ over 60 centre-points. The results are $\hat{\Delta}_c = 0.154$ for liver tumours and $\hat{\Delta}_c = 0.0.250$ for MS lesions. We translate a centre point estimate according to equation 8.3 by first drawing an angle θ from a uniform distribution and compute $\Delta x = \hat{\Delta}_c \bar{r} \cos(\theta)$and $\Delta y = \hat{\Delta}_c \bar{r} \sin\theta$. Figure 8.2 shows the effect of translations $\{x_c, y_c\} \rightarrow \{x_c, y_c\} \pm \{\Delta x, \Delta y\}$, on a perfect circle (a). Each translation adds a sinusoidal trend to the centre ($\bar{r}$) of a radial series (b). It follows that centre point perturbation affects any time series in the space of (b) by a periodic offset $\Delta r(\theta)$ given by

$$\Delta r(\theta) = \Delta x \cos(\theta) + \Delta y \sin(\theta). \tag{8.4}$$

When modelling centre point uncertainty by translations $\{\Delta x, \Delta y\}$ we add the sinusoidal trend in equation 8.4 to a generated time series. This scheme reduces computation time, compared to generating a series, transforming into image space, and then translating by $\{\Delta x, \Delta y\}$

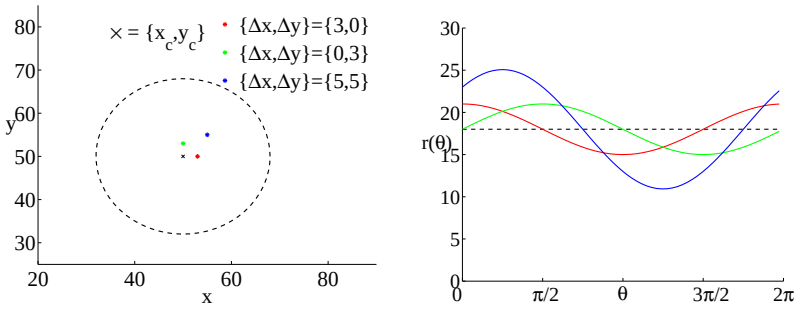


Figure 8.2: Effect of perturbing the centre-point on a constant radial time series.

8.1.3 Bayesian formulation

The generative segmentation algorithms adopt a Bayesian formulation so that an optimisation scheme ($\mathcal{C}_5$) estimates the maximum *a-posteriori* probability (MAP) solution $\mathbf{Q}_{\text{star}}^{\text{MAP}}$. The MAP solution is the shape that maximises the posterior probability given by

$$\Pr(\mathbf{Q}|\mathcal{D}) = \Pr(\mathcal{D}|\mathbf{Q})\Pr(\mathbf{Q}), \tag{8.5}$$

where $\Pr(\mathbf{Q})$ is the shape prior and $\Pr(\mathcal{D}|\mathbf{Q})$ is the data likelihood. Recalling $\mathbf{Q}_{\text{star}} = \{\mathbf{r}, \bar{r}, \mathbf{x}_c\}$ we can write

$$\Pr(\mathbf{Q}) = \Pr(\mathbf{r})\Pr(\mathbf{x}_c)\Pr(\bar{r}) \tag{8.6}$$

where $\Pr(\mathbf{r})$ is a probabilistic 'score' from equation 7.6 or 7.16, $\Pr(\mathbf{x}_c)$ comes from equation 8.3 and $\Pr(\bar{r})$ is estimated for a given image as described later. We construct the data likelihood by repeating equation 8.2 for all $\theta_i \in \theta$, giving

$$\Pr(\mathcal{D}|\mathbf{Q}) = \Pr(\{\mathbf{p}^{\text{on}}, \mathbf{p}^{\text{off}}\}|\mathbf{Q}), \tag{8.7}$$

where $\mathbf{p}^{\text{on}} = \{p_0^{\text{on}}, \ldots p_i^{\text{on}}, \ldots p_{N-1}^{\text{on}}\}$ and $\mathbf{p}^{\text{off}} = \{p_0^{\text{off}} \ldots p_i^{\text{off}}, \ldots p_{N-1}^{\text{off}}\}$. This sets up the general Bayesian formulation for all time series shape models used in segmentation. Sections 8.4.1 and 8.5 present specific methods for Bayesian MAP estimation using generative models. First, the next section uses the discriminative shape models for shape regularisation.

8.2 Shape Regularisation for Segmentation

This section presents methods of shape regularisation using the Langevin model and discusses regularisation with the GP model. We start with a simple deformable contour model and incorporate discriminative shape models into the objective function. Section 8.2.1 develops the theory for a segmentation framework and section 8.2.2 demonstrates its performance on synthetic images to observe the role of the shape prior.

8.2.1 Radial active contour model (RACM): a simple framework

The deformable contour model is adapted from the classical 'snake' of Kass *et al*, [5] for use with the shape models. We refer to this DCM as the *Radial Active Contour Model* (RACM), which is charac-

terised by the choice of contour parametrisation ($\mathcal{C}_1$), objective function ($\mathcal{C}_4$) deformation mechanism ($\mathcal{C}_5$) and optimisation scheme ($\mathcal{C}_6$).

The contour parametrisation ($\mathcal{C}_1$) is the radial time series 7.1(a) or 7.1(b). We demonstrate for the star-shaped case (7.1(b)) for convenience, as it enables us to work with the radial profile image model introduced above. The RACM extends to the generalised parametrisation (7.1(a)), by using a consistent image model derived from the image frame.

The objective function ($\mathcal{C}_4$) is an energy functional, combining the shape model with elements of the observation model $\mathcal{D}$. Upon estimation of the region centre $\mathbf{x}_c$, we calculate vectors of N means $\hat{G}$ and N variances σ^g from the radial profile model above, where for simplicity the observations are gradient magnitude. We use these to define an *image energy* term given by

$$E_{\text{image}}(\mathbf{r}) = \sum_{i=0}^{N-1} \exp\left[-\frac{(r_i - \hat{g}_i)^2}{(\sigma_i^g)^2}\right]. \tag{8.8}$$

We also define a *shape energy* term, which is simply the normalised log probability used for shape scoring in the previous chapter

$$\begin{aligned} E_{\text{shape}}(\mathbf{r}) &= S_{\text{LAN}} \qquad \text{for Langevin, and} \\ E_{\text{shape}}(\mathbf{r}) &= S_{\text{GP}} \qquad \text{for GP,} \end{aligned} \tag{8.9}$$

where S_{LAN} and S_{GP} are given by equations 7.6 and 7.16 respectively. Finally the contour energy is defined by

$$E = \alpha E_{\text{image}} + (1 - \alpha) E_{\text{shape}}, \tag{8.10}$$

where α is a parameter that controls the relative influence of image and shape model.

The deformation mechanism ($\mathcal{C}_5$) and optimisation scheme ($\mathcal{C}_6$) combine stochastic sampling with a greedy algorithm. Greedy algorithms such as the one described for snakes in section 2.1.2 are attractive for their simplicity. By replacing the greedy search with stochastic sampling we introduce the benefits of a stochastic framework as noted in section 2.5, as well as avoiding the case where a greedy algorithm enters an oscillatory state rather than converging.

The RACM is initialised with a noisy circle centred on $\mathbf{x}_c$, with radius $\bar{r}$ estimated by the mean of $\hat{\mathbf{g}}$. The algorithms proceed by perturbing successive points r_i (in the Langevin case) or whole series $\mathbf{r}$ (in the GP case). In the Langevin case we draw each r_i' in turn from $\mathcal{N}(r_i, 1)$, while in the GP case we draw a series $\mathbf{r}'$ from $\mathcal{N}_{\text{N}}(\mathbf{r}, \mathbf{I})$. In each case the perturbation is 'accepted' if it causes a reduction in global energy E. The RACM terminates after a fixed number of iterations or, in the Langevin case, when all radii in a series are perturbed without acceptance, giving no change in energy between successive iterations.

8.2.2 Demonstration of the Langevin RACM

Next, we demonstrate the RACM, used with the Langevin model and different weightings α. Figure 8.3 shows the results using the Langevin model for a synthetic images low signal-to-noise ratios (SNR). The boundary is given by a ground truth liver tumour, which was omitted from the training set. We repeat segmentation with α decreasing from 1, where the shape prior is ignored, to 0.5, where image

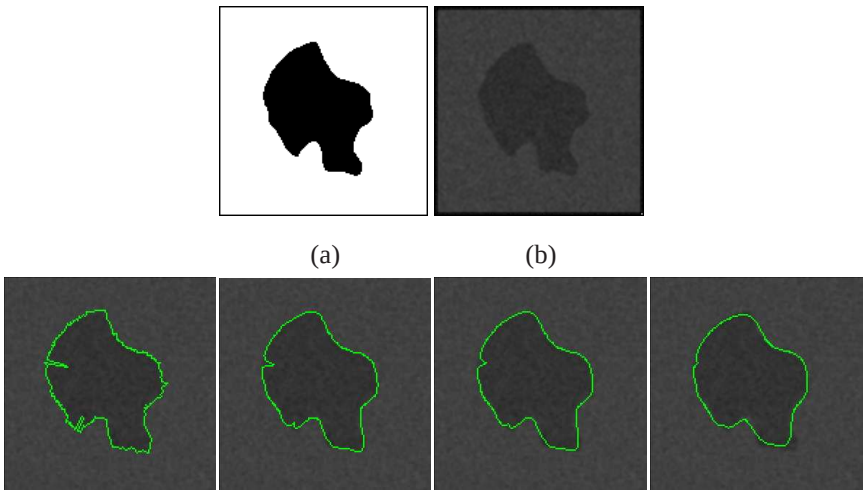


Figure 8.3: (a) Binary image showing ground truth liver tumour shape. (b) Synthetic liver tumour image with SNR=1.84. (c) - (f) Results of RACM segmentation, (green contours) using Langevin regularisation with (c) $\alpha = 1.0$, (d) $\alpha = 0.85$, (e) $\alpha = 0.75$ and (f) $\alpha = 0.5$.

and shape priors are equally weighted. Panel (c) shows the affect of image noise in the absence of shape priors, where points of high gradient resulting from image noise cause jagged boundaries. Panels (d) and (e) seem to give a good balance between image and shape models while the smoother contour in panel (f) suggests that the RACM is over-constrained when image and shape energies are equally weighted ($\alpha = 0.5$).

We also implemented the RACM with regularisation by the GP model and found the approach to be impractical for two reasons. First, GP regularisation is very slow, as scoring shapes by equation 7.16 requires inverting a $N \times N$ matrix at each iteration of the RACM. In practice the GP RACM barely converges after 500 iterations, which can take several minutes. Second, it is not clear when the RACM has converged under GP regularisation. The algorithm iteratively considers proposals in the form of complete contours, and must be able to reject a complete contour at any iteration. This means that an energy change of zero is not an appropriate termination criterion.

The use of GP models for shape regularisation may be suited to non-interactive tasks such as image reconstruction or registration, which can be performed off-line.

In summary, we have presented a general framework for segmenting star-shaped regions with Langevin regularisation, which naturally extends to the generalised Langevin model and parametrisation. We also state that GP regularisation is possible but defer this for future work.

8.3 A Generalised Framework for Interactive Segmentation using Generative Shape Models

We have just seen how image and shape models can be used together in an ACM framework, where a contour is deformed stochastically ($\mathcal{C}_5$), and an optimisation scheme minimises an energy functional that includes a term for the shape prior. Next we introduce a different approach to segmentation, where generative shape models replace the deformation mechanism and an *a posteriori* probability replaces the energy functional. This section presents the general framework, which is independent of the choice of shape model. We state in general terms, and with reference to the following sections 8.4 and 8.5, how to proceed using the Langevin and GP shape models. This places the subsequent sections in context, as well as contributing to the wider field of segmentation by formalising a general framework

The framework comprises the general components in the left hand column of table 8.1, where the right hand column introduces the corresponding Langevin and GP method to be detailed in the next 2 sections.

8.4 Generative Langevin Models for Segmentation

Section 5.1.2 described how a Langevin series is simulated by the numerical solution of a stochastic differential equation (SDE). In the context of shape modelling we desire a similar scheme to generate shapes from a model. The resulting shapes can serve as proposals or 'hypotheses' in probabilistic segmentation frameworks. This section introduces adaptations to the Euler-Maruyama scheme, designed to generate appropriate series. Subsection 8.4.1 adapts the Euler-Maruyama scheme for solving SDEs, making it appropriate for shape generation by generating closed contours and addressing issues concerning the centre point $\mathbf{x}_c$ and the discrete nature of training data. Subsection 8.4.2 combines particle-filtering with the generative models as a novel approach to data assimilation and 8.4.3 presents the segmentation framework.

8.4.1 Euler-Maruyama scheme for shapes

We start by writing the Euler-Maruyama scheme in polar coordinates

$$r(\theta + \mathrm{d}\theta) = r(\theta) + \mathrm{d}\theta \times A(r(\theta)) + \sqrt{\mathrm{d}\theta} \times B(r(\theta))\omega(\theta), \tag{8.11}$$

which we initialise with a small value $r(0) = 0.01$. Next we choose the integration time step $\mathrm{d}\theta$ via the following observations.We noted in section 5.1.2, that no single choice of integration time step is optimal for all Langevin models. In practice we find that the models are not too sensitive to this parameter for a range of small values. There is an upper limit, however, as larger values introduce chaotic behaviour to the stochastic process. To explain this behaviour we note that the well-known 'logistic equation' [277] which models chaotic fluctuations of a population x as $x(t+1) = kx(t)(1 - x(t)))$, is a special case of equation 8.11. We choose $\mathrm{d}\theta \lesssim 0.5$, which works well in practice.

Next we incorporate the effect of centre point perturbation in equation 8.4. The Euler-Maruyama

General component proposed	Examples we present
(1) any SSM that defines a closed contour $\mathbf{Q}$ as an instance of a contour representation , and uses machine learning methods.	Langevin and GP SSMs from chapter 7 above, with $\mathbf{Q} = \mathbf{Q}_{\text{star}} = \{\mathbf{r}, \mathbf{x}_c, \bar{r}\}$.
(2) any model that derives a data likelihood $\mathcal{D}$ from information in an image	The 'radial profile' model introduced in section 8.1.1 above .
(3) an interactive method of initialisation , which provides as much information to the shape model as possible from as simple an interaction as possible .	Mouse curser click in the centre of a region. Gives estimate $\mathbf{x}'_c$ for use with the uncertainty model in section 8.1.2. Also used to extract the radial profile model from the image and in turn estimate the SSM parameter $\bar{r}$.
(4) a generative mechanism, which is capable of drawing samples from the prior distribution $\Pr(\mathbf{Q})$ after the model is trained	Generative Langevin SSM (section 8.4.2) or generative GP SSM (section 8.5.1).
(5) a method of constraining the generative model to draw samples from the posterior distribution $\Pr(\mathbf{Q}\|\mathcal{D})$.	Langevin method using the generative model as a globally adaptive prior in *step-wise* particle filtering (section 8.4.2). GP method using probabilistic regression technique where the image provides noisy observations (section 8.5.1).
(6) an optimisation scheme capable of estimating the maximum *a posteriori* probability (MAP) solution $\mathbf{Q}^{\text{MAP}}$, accounting for the inaccuracy of user initialisation.	Maxim. $\Pr(\mathcal{D}\|\mathbf{Q}) = \Pr(\Pr(\mathcal{D}\|\Pr(\mathbf{r})\Pr(\mathbf{x}_c)\Pr(\bar{r}))$, accounting for uncertainty on the user-initialised centre $\mathbf{x}'_c$. Done by estimating the MAP solution directly from (5) and combining with *shape-wise* particle filtering in sections 8.4.3 (Langevin) and 8.5.2 (GP) .
(7) an interactive method of post editing, which works with the shape model as closely as possible.	Enabling repeated initialisation, and boundary-based correction procedures for the Langevin (section 8.4.3.1) and GP contours (section 8.5.2.1), where the GP method refines the shape prior and re-calculates the MAP estimate .

Table 8.1: Components of a generalised framework for interactive segmentation using generative SSMs. The components are listed here along with the examples presented for Langevin and GP frameworks.

scheme becomes

$$r(\theta + \mathrm{d}\theta) = r(\theta) + \mathrm{d}\theta \times A(r(\theta)) + \sqrt{\mathrm{d}\theta} \times B(r(\theta))\omega(\theta) + \Delta x \cos(\theta) + \Delta y \sin(\theta), \qquad (8.12)$$

where $\{\Delta x, \Delta y\}$ are drawn from $\mathcal{N}_2(\mathbf{x}'_c, \sigma^{\mathbf{x}_c})$ during optimisation in the algorithms introduced below.

The next two subsections address certain issues regarding the generation of closed contours (8.4.1.1), and discrepancies between a prior model and generated series resulting from the discretisation of training data in $\{x, y\}$ coordinates (8.4.1.2).

8.4.1.1 Generating Closed Contours

The Langevin models were partly motivated by conclusions made in section 6.4, that boundary tracking methods are prone to self intersection and lack satisfactory methods of loop closing. The star-shaped parametrisation naturally rules out self-intersecting contours by asserting monotonically increasing angle θ. However, Langevin series allow discontinuities at the start/end of a 2π cycle due to the Markov property. A discontinuity arises from a net displacement $|r_{N-1} - r_0| > 0$ over a period of N points. However, due to the natural fluctuations in the time series, discontinuities are only apparent if $|r_{N-1} - r_0| > B(r_{N-1}, \mathbf{b}) \approx B(r_0, \mathbf{b})$, i.e. the difference between first and last radii is greater than the magnitude of the noise term local to the radius of the 'join'. As such we can produce a pseudo-periodic series by allowing the magnitudes of r_{N-1} and r_0 to differ within a 'tolerance' of $|r_{N-1} - r_0| \leq B(r_{N-1}, \mathbf{b})$.

To generate closed contours we run the Euler-Maruyama integration for $N' \geq N - 1$ iterations, terminating as soon as $|r_{N'} - r_{N'-N}| \leq B(r_{N'}, \mathbf{b})$. Two observations reveal the efficiency of the proposed algorithm. First, the algorithm need never store more than N points at a time as, by using a 'list'-type data structure, we remove the zeroth point from the list each time the Nth point is added. Second, the existence of stable regions in the radius space (governed by the drift function) increases the likelihood that two points r_i and r_j, of arbitrary separation $j - i$, are at similar radii. However, care should be taken if a model has stable regions at either side of the zero-centre. In these cases, a series might satisfy the termination criterion after spending N iterations in the positive (or negative) half of state space. The resulting series would not occupy a zero-mean field, affecting estimates of the scale parameter $\bar{r}$. To preserve the zero-mean field we sum the *sign* of series values over N points and only accept a closed loop that has a total sign within the range ± 0.2, chosen empirically. The joint termination criterion for generating model-consistent closed contours is therefore given by

$$|r_{N'} - r_{N'-N}| \leq B(r_{N'}, \mathbf{b}) \qquad \text{and} \qquad \sum_{i=N'-N}^{N'} \text{sign}(r_i) < 0.2. \tag{8.13}$$

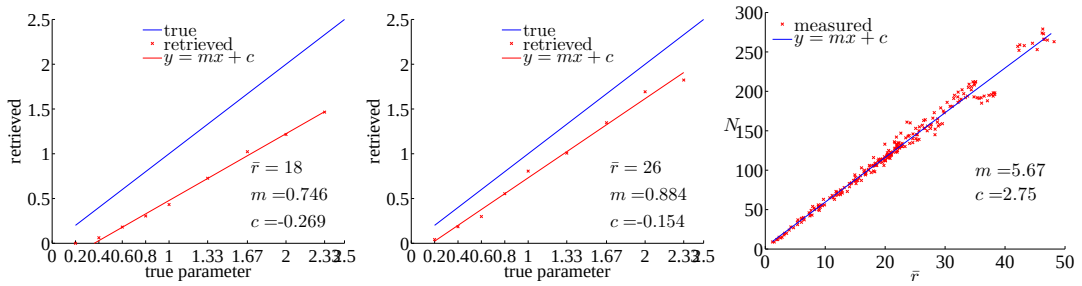
The use of the joint termination criteria in equation 8.13 raises two issues. First, excepting the special case where N' is an integer number of N, the termination criteria result in an angular offset between the 'start' of the series and the presumed angle $\theta = 0$ (horizontal in the image frame). While the prior shape model is rotation invariant, so $\theta = 0$ is arbitrary, the use of generative models with image observations below demands that the initial angle in equation 8.12 corresponds to the 'start' of the radial profile model. The segmentation algorithms recognise and correct for this offset.

Second, the need for $N' \geq N$ iterations to generate a series of N points could slow down the algorithm, in theory to speeds below what is practical in an interactive framework. The probability of termination is governed by the shape prior and, when used in segmentation, the image model that constrains series generation (below). This type of constraint is expected to speed up shape generation in

a similar way to the probabilistic loop closing algorithm in section 6.1.3.1. In practice the criteria are met quickly and any delay is not noticeable.

8.4.1.2 Compensating for discretisation errors

Because training shapes are originally defined on a pixel grid, the state space r of training series is discretised. This leads to an inconsistency between the dynamical properties learned from data and those of a generated series. To investigate the inconsistency we trained models on synthetic data generated by known models, where we expect to retrieve the known parameters. We repeat for two types of synthetic data. In the first case we generate series, transform these into shapes of chosen $\bar{r}$ on a discrete $\{x, y\}$ grid, then re-sample the shapes in the same way that radial time series are derived from medical ROI contours. In the second case we generate series and leave them un-touched. We find that models trained on raw series retrieve the 'true' parameters while models trained on series derived from shapes give errors. Figure 8.4 (a)/(b) illustrate this effect for a simple model using a cubic drift function $A(r(\theta), \mathbf{a}) = a_0(1-r(\theta)^3)$ with a single parameter $\mathbf{a} = \{a_0\}$. Figure 8.4 shows that the discrepancy is systematic and apparently



(a) (b) (c)

Figure 8.4: Effect of state space discretisation on shape dynamics. Panels (a) and (b) show the offset between the 'true' parameter in the drift function of a Langevin model used to generate synthetic shapes, and the parameter retrieved from 500 of these shapes by the direct estimation method. The offset is slightly different for shapes generated with (a) $\bar{r} = 18$ and (b) $\bar{r} = 26$. Panel (c) shows the relationship between scale parameter $\bar{r}$ and the number of pixels N in a training contour.

linear. Comparing plots (a) and (b) reveals that the discrepancy also depends on the scale parameter $\bar{r}$, as the state space is more finely discretised further from a region's centre.To correct for discrepancies between those parameters we learn and those that would reproduce consistent boundary dynamics in the generative model, we store parameter sets in vectors of the form $\{a_0, \ldots a_p, b_0, \ldots b_p, \bar{r}\}$ and compute a linear mapping between learned and 'true' sets. For each combination of drift and diffusion function we vary $\{a_0, \ldots a_p, b_0, \ldots b_p, \bar{r}\}$ so that mappings generalise over a range of parameter choices. The result allows us to calibrate any trained model for the purpose of shape generation.

Finally we need to down-sample the high resolution series before transforming to the image frame $\{x, y\}$. As the training data were up-sampled to a common, high resolution, series generation must also occur at this resolution. This would lead to too high boundary resolution in all but the largest shapes (greatest $\bar{r}$). We down-sample each series to contain N points, where N is chosen from a linear

relationship between N and $\bar{r}$ in the training data. Figure 8.4 (c) shows this relationship in the case of liver tumour boundaries.

Figure 8.5 demonstrates the use of the generative shape models for creating shape instances (irrespective of an image). The figure verifies the success of the loop closing procedures, as well as revealing the affect of calibration and changing drift/diffusion parameters. Examples demonstrate affect of chang-

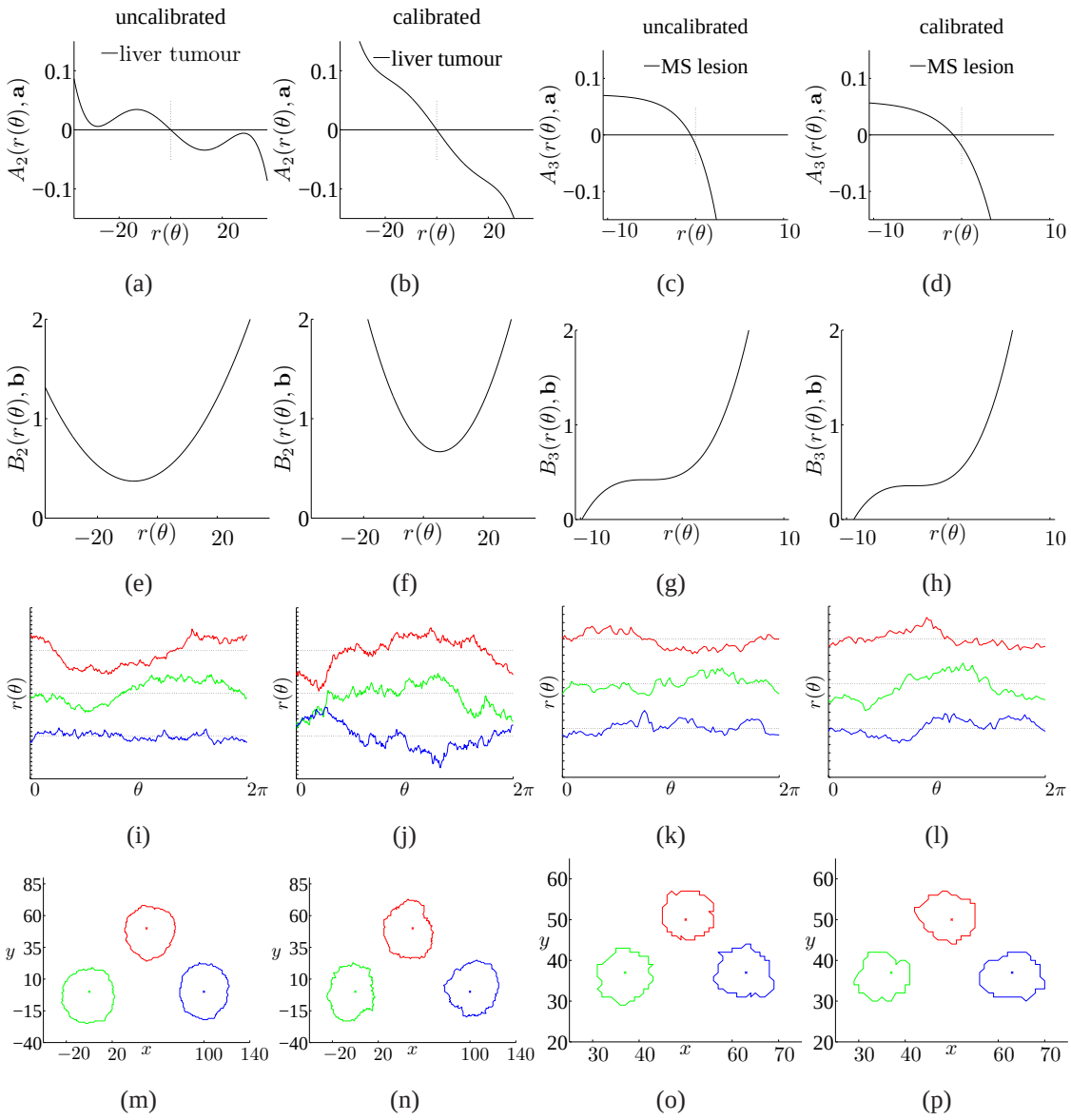


Figure 8.5: Example instances from generative Langevin shape models. *Top row*: the drift functions (a) after parameter estimation and (b) after calibration, for the liver tumour data, and (c)/(d) the same for the MS lesion data. *Second row*: the corresponding diffusion functions (e)/(f) for liver tumours and (g)/(h) for MS lesions. *Third row*: three instances of series generated by each model. *Bottom row*: the corresponding shapes.

ing the drift and diffusion parameters, on generated series/shapes. We also note that discontinuities are not visually noticeable.

8.4.2 Data assimilation

Before we can estimate $\mathbf{Q}_{\text{star}}^{\text{MAP}}$ we first need a method of drawing closed contours from the posterior distribution $\Pr(\mathbf{Q}|\mathcal{D})$ (equation 8.5). This means generating closed contours that not only agree with the shape model as above, but also incorporate observations from the image. Incorporating observations is equivalent to *data assimilation*, which we described in section 5.1.3 and noted that this is the subject of ongoing research [240, 241] in the case of Langevin simulation. We present a data assimilation method inspired by the particle filters in [43]. The result is an example of boundary tracking according to the definition in 2.2, but which tracks a whole boundary to produce a closed contour with global shape constraints.

We further adapt the Euler-Maruyama scheme above so that the solution of the SDE is equivalent to the iterative computation of posterior densities as in section 2.2.1. Rewriting the jetstream equation 6.1 for the time series models gives

$$\Pr(r_{i+1}|r_i, \mathbf{a}, \mathbf{b}, \mathcal{D}) \propto \Pr(r_i|\mathbf{a}, \mathbf{b}, \mathcal{D}) \times q(r_{i+1}|r_i, \mathbf{a}, \mathbf{b}) \times l(\mathcal{D}|r_{i+1}), \tag{8.14}$$

where q and l now denote a global shape prior and data likelihood given by

$$\begin{aligned} q(r_{i+1}|r_i, \mathbf{a}, \mathbf{b}) &= \mathcal{N}(r_i - A(r_{i+1}, \mathbf{a}), B(r_{i+1}, \mathbf{b})) \qquad \text{and} \\ l(\mathcal{D}|r_{i+1}) &= \frac{p_{i+1}^{\text{on}}}{p_{i+1}^{\text{off}}}. \end{aligned} \tag{8.15}$$

Note also that 8.14 shares the Markov property of Langevin models (equation 5.2).

For a fixed centre point $\mathbf{x}_c$, the algorithm generates a set of K series $\mathbf{r}^k$, $k = 0, \ldots K-1$ as follows. At each step θ_i we draw K predictions for $r_{i+1}^{k=0,\ldots K-1}$ solving the SDE in equation 8.12) K times for a fixed r_i. We then assign weights $w_{i+1}^{k=0,\ldots K-1}$ to each prediction, given by

$$w_{i+1}^k = \alpha \frac{p_i^{\text{on}}}{p_i^{\text{off}}} + (1-\alpha)\mathcal{N}(r_i - A(r_{i+1}, \mathbf{a}), B(r_i, \mathbf{b})), \tag{8.16}$$

where α controls the relative influence of shape and image priors as in the regularisation example in section 8.2. The weights w form a discrete approximation of the posterior $\Pr(r_{i+1}|\mathbf{a}, \mathbf{b}, \mathcal{D})$, specific to θ_i. We perform *step-wise* importance sampling, by selecting K points with replacement from the posterior. Repeating for N steps results in K separate series that can be mapped into image space. Finally we take the single radial time series (closed contour) corresponding to the points of maximum weight w_i^*.

8.4.3 Probabilistic algorithm

The techniques above modify the numerical integration scheme to incorporate the data likelihood by a *step wise* data assimilation technique. The resulting contours estimate the region boundary for a fixed centre point $\mathbf{x}_c$ and scale parameter $\bar{r}$ in equation 8.6. To estimate the (MAP) solution $\mathbf{Q}_{\text{star}}^{\text{MAP}} = \underset{\mathbf{Q}_{\text{star}}}{\arg\max} \Pr(\mathbf{Q}_{\text{star}}|\mathcal{D})$, where $\mathbf{Q}_{\text{star}} = \{\mathbf{r}, \mathbf{x}_c, \bar{r}\}$, we simultaneously seek $\mathbf{x}_c$ and $\bar{r}$ in the optimisation scheme. (the simultaneous optimisation of $\mathbf{x}_c$ is in common with the approach taken in [73]). We extend the particle filter approach to sample M complete contours from the posterior distribution.

We draw M combinations of centre points $\mathbf{x}_c$ and scale parameter $\bar{r}$ from the distributions $\Pr(\mathbf{x}_c) = \mathcal{N}_2(\mathbf{x}'_c, \Delta_c)$, centred on the user initialisation $\mathbf{x}'_c$, and $\Pr(\bar{r}) = \mathcal{N}(\hat{\bar{r}}, \sigma^2_{\bar{r}})$, where $\hat{\bar{r}}$ comes from the mean of $\hat{g}_i$ estimated from the local image gradient and $\sigma^2_{\bar{r}}$ could either come from the image as the mean of σ^g_i or chosen empirically. For each combination we generate a closed contour with data assimilation as described in section 8.4.2 and assign a weight to each closed contour, given by

$$W_{\mathbf{Q}_{\text{star}}} = \prod_{i=0}^{N-1} w^*_i \tag{8.17}$$

where w^*_i are the maximum step-wise weights as described above. We refer to this procedure as *shape wise* particle filtering, whereby each particle is a closed contour. The combination of step-wise and shape-wise particle filtering results in a nested algorithm, which is fast enough in practice for use in real-time segmentation.

8.4.3.1 Interactivity and post editing

The Langevin framework has two modes of interaction, for initialisation and post editing. Figure 8.6 demonstrates the use of this tool. A user initialises the segmentation by giving the centre point estimate $\mathbf{x}'_c$ inside the region of interest. Upon initialisation the software immediately displays the contour given by the MAP solution above. The user can repeat the initialisation any number of times, which might give different contours. The user can edit the displayed contour by dragging any of the points r_i in the contour model that miss the desired region boundary. It is possible that contour points are too close together, meaning that a large number would need to be dragged onto a small section of the boundary. To alleviate this problem we allow the user to adjust the boundary resolution using a slide-bar. This adjustment is reversible.

8.5 Generative Gaussian Processes for Segmentation

This section adapts GP models to generate model shapes (hypotheses) and, in section 8.5.1, to incorporate the data likelihood from the radial profile observation model above (section 8.1.1).

Whereas the Langevin model required numerical procedures to ensure the generation of closed contours, GP models facilitate analytical closed contour models. This is because, unlike the Markovian case of Langevin models, GP models encode the global constraint of correlating a point $r(\theta)$ with $r(\theta + 2\pi)$ by using a periodic kernel function. We use equation 7.11 introduced in section 7.3.1.

GP models readily generate samples from the prior as described in section 5.2.2. In the case of radial time series this means evaluating

$$\mathbf{r}' = \mu + \mathbf{A}\mathbf{z}. \tag{8.18}$$

As with the Langevin models above, we use a linear relationship to calibrate the generative model with respect to the parameter estimation, and down-sample a generated series to give a suitable boundary resolution for the corresponding scale parameter. Figure 8.7 shows the chosen kernel function for different parameters $\mathbf{a} = \{a\}$, along with the corresponding radial time series and shapes.

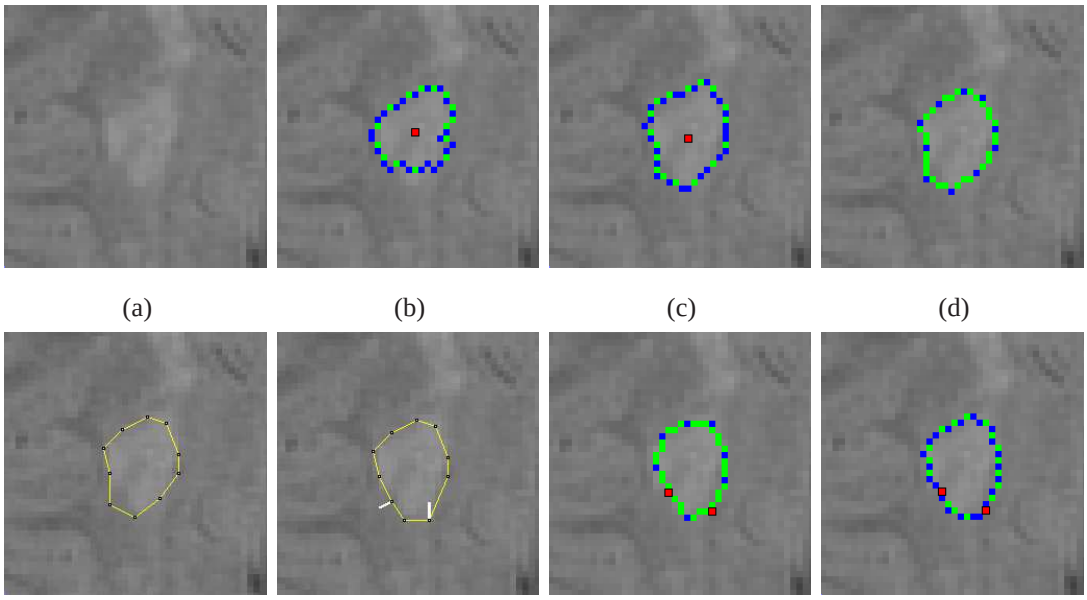


Figure 8.6: Interactive contouring with the Langevin SSM tool. (a) Close-up of a MS lesion in an axial PD weighted MR image. (b) A contour (MAP estimate) shown in blue/green after selection of an initial estimate of the centre point $\mathbf{x}_c'$ (red). (c) An alternative initial contour shown after re-initialisation with a new estimate $\mathbf{x}_c'$. The same contour as in (b), shown after the user has reduced the resolution of those boundary points (blue) that can be 'dragged'. (e)/(f) The dragging mode before/after two boundary points are moved. White lines are added here to highlight the translation of the dragged points, but are not shown to the user during run-time. (g)/(h) The final contour before/after returning to the original boundary resolution.

8.5.1 Conditioning the prior

This section shows how to directly generate samples from the posterior over contours $\Pr(\mathbf{Q}_{\text{star}}|\mathcal{D}) \propto \Pr(\mathcal{D}|\mathbf{Q}_{\text{star}})\Pr(\mathbf{Q}_{\text{star}})$ where $\Pr(\mathbf{Q}_{\text{star}})$ is the shape prior and $\Pr(\mathcal{D}|\mathbf{Q}_{\text{star}})$ is the data likelihood. We follow the method of conditioning the prior on 'noisy observations' as described in section 5.2.3. This has the same role as the step-wise particle filter in the Langevin case but has two key differences. First, unlike the Markovian case, the observation model constrains the whole radial time series in 'one-shot'. Second, the angles θ_i at which observations are made need not belong to the vector of inputs θ. We can use any number of observations at arbitrary angles.

We define a noisy observation at angle θ_i using the radial profile model. We start with the Gaussian model of the gradient magnitude, i.e. $1 + p_i^{\text{off}}(r)$ where p_i^{off} is from equation 8.2 in the radial profile model. The noisy observation for the ith observation angle is given by $\mathcal{N}(\hat{g}_i, \sigma_i^g)$.

For an arbitrary number N of observationse store a vector of means $\hat{g}_i$ and a matrix of variances $(1 - p_i^{\text{on}}(r))\sigma_i^g\mathbf{I}$, where $\mathbf{I}$ is a $N \times N$ identity matrix. We construct a vector θ' of length $N' = N + N^*$ by concatenating the vector of observation inputs θ with the vector of N^* inputs for which we do not

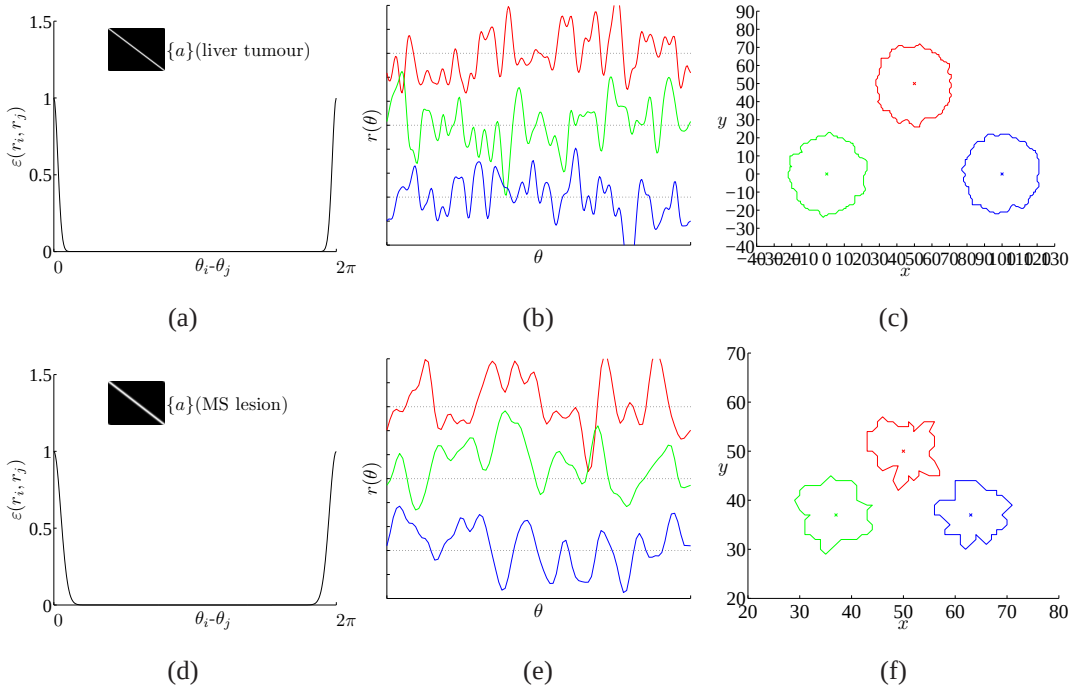


Figure 8.7: Generative GP model trained on liver tumours (top row) and MS lesions (bottom row). Panels (a)/(d) show the kernel functions with a representation of the covariance matrix (inset). Panels (b)/(e) each show three instances of a generated series and (c)/(f) show the corresponding shapes with typical scale factor $\bar{r}$

have observations, denoted θ^*. This new vector $\theta' = \{\theta, \theta^*\}^T$ has a covariance matrix of the form

$$\begin{aligned}\boldsymbol{\Sigma}(\theta', \theta') &= \begin{pmatrix} \varepsilon(\theta_0, \theta_0) & \dots & \varepsilon(\theta_0, \theta^*_{N'-1}) \\ \vdots & \ddots & \vdots \\ \varepsilon(\theta^*_{N'-1}, \theta_0) & \dots & \varepsilon(\theta^*_{N'-1}, \theta^*_{N'-1}) \end{pmatrix} \\ &= \begin{pmatrix} \boldsymbol{\Sigma}_{\theta\theta} + \sigma^2\mathbf{I} & \boldsymbol{\Sigma}_{\theta\theta^*} \\ \boldsymbol{\Sigma}_{\theta^*\theta} & \boldsymbol{\Sigma}_{\theta^*\theta^*} \end{pmatrix},\end{aligned} \tag{8.19}$$

where $\boldsymbol{\Sigma}_{\theta\theta}$ denotes a $N \times N$ sub matrix, $\boldsymbol{\Sigma}_{\theta\theta^*}$ is a $N \times N^*$ sub matrix and so on. These sub-matrices are used to form the posterior model defined by

$$\mathbf{r}' = \mathcal{N}_N(\mu_{\text{post}}, \boldsymbol{\Sigma}_{\text{post}}), \tag{8.20}$$

where

$$\mu_{\text{post}} = \mu + \boldsymbol{\Sigma}_{\theta^*\theta}[\boldsymbol{\Sigma}_{\theta\theta} + \sigma^2\mathbf{I}]^{-1}(\hat{\mathbf{G}} - \mu) \tag{8.21}$$

and

$$\boldsymbol{\Sigma}_{\text{post}} = \boldsymbol{\Sigma}_{\theta^*\theta^*} - \boldsymbol{\Sigma}_{\theta^*\theta}[\boldsymbol{\Sigma}_{\theta\theta} + \sigma^2\mathbf{I}]^{-1}\boldsymbol{\Sigma}_{\theta\theta^*}. \tag{8.22}$$

We could draw samples from the posterior by constructing a N-dimensional vector $\mathbf{Z}$, where z_i is from $\mathcal{N}(0, 1)$, then evaluating $\mathbf{r}^* = \mu_{\text{post}} + \mathbf{AZ}$. This is equivalent to replacing μ and $\boldsymbol{\Sigma}$ in equation 8.18 with equations 8.21 and 8.22 respectively. However, these samples are of limited use to the present

framework. Whereas in the Langevin case we estimate the MAP solution (for a fixed centre-point and scale parameter) from such samples, the GP model allows to calculate the MAP estimate directly from the posterior model. This is because each $\Pr(r_i|\theta_i)$ is a Gaussian distribution, so the MAP estimate is equivalent to the posterior mean calculated at all inputs, given by

$$\mathbf{Q}^{\text{MAP}} = \mu_{\text{post}} = \mu(\theta^*) + \mathbf{\Sigma}(\theta^*, \theta)\left[\mathbf{\Sigma}(\theta, \theta) + \sigma_{\mathbf{g}}^{\mathbf{2}}\mathbf{I}\right]^{-1}\left(\hat{\mathbf{g}} - \mu(\theta)\right) \tag{8.23}$$

where vectors $\hat{\mathbf{g}}$ and $\sigma_{\mathbf{g}}$ contain the observations from the radial profiles described in section 8.1.1 and $\mu(\theta^*)$ and $\mu(\theta)$ is the mean function evaluated at the inputs corresponding to unobserved and observed data respectively.

8.5.2 Probabilistic algorithm

The estimate in equation 8.23 has a unique solution for a given centre-point and estimated $\bar{r}$. The present segmentation framework has three remaining requirements. First, we require the MAP estimate $\mathbf{Q}_{\text{star}}^{\text{MAP}} = \underset{\mathbf{Q}_{\text{star}}}{\arg\max}\ \Pr(\mathbf{Q}_{\text{star}}|\mathcal{D})$ where $\mathbf{Q}_{\text{star}} = \{\mathbf{r}, \mathbf{x}_c, \bar{r}\}$. Second, we require the use of full observation model incorporating the local boundary direction in the likelihood ratio $p_i^{\text{on}}(r)/p_i^{\text{off}}(r)$. Third, we require control over the relative influence of image and shape models. The chosen algorithm achieves all three of these requirements by using a nested particle filter as follows.

We draw M combinations of centre points $\mathbf{x}_c$ and scale parameter $\bar{r}$ from the distributions $\Pr(\mathbf{x}_c) = \mathcal{N}_2(\mathbf{x}'_c, \Delta_c)$, centred on the user initialisation $\mathbf{x}'_c$, and $\Pr(\bar{r}) = \mathcal{N}(\hat{\bar{r}}, \sigma_{\bar{r}}^2)$, where $\hat{\bar{r}}$ comes from the mean of $\hat{g}_i$ estimated from the local image gradient and $\sigma_{\bar{r}}^2$ could either come from the image as the mean of σ_i^g or chosen empirically. For each combination we compute the posterior mean by equation 8.23, for use as M shape-wise particles. We weight each shape by

$$W_{\mathbf{Q}_{\text{star}}} = \alpha\frac{1}{N}\sum_{i=0}^{N-1}\frac{p_i^{\text{on}}(r)}{p_i^{\text{off}}(r)} + (1-\alpha)S_{\text{GP}} \tag{8.24}$$

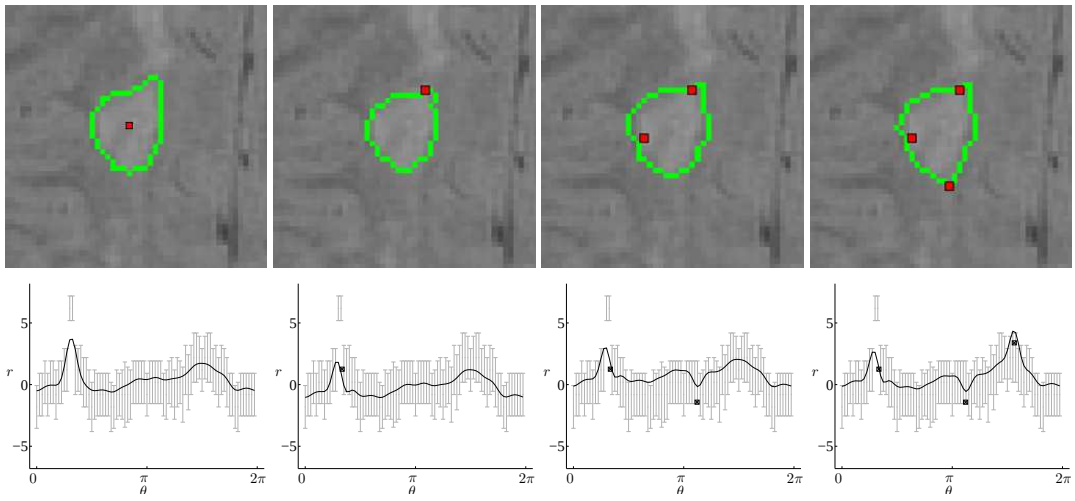
where $p_i^{\text{on}}(r)$ and $p_i^{\text{off}}(r)$ are from the observation model (equation 8.2), S_{GP} is the Gaussian process shape score in equation 7.16 and α is the relative weighting between image and shape models.

8.5.2.1 Interactivity and post editing

As in the Langevin case, the GP framework involves two modes of interaction for initialisation and post editing. Initialisation estimates the centre point $\mathbf{x}_c$ as before. The post editing procedure is new, and makes more efficient use of interactions than the point-dragging mode in the Langevin framework. The ability to interact closely with an underlying shape prior should intuitively reduce the demand on the user and has been shown to benefit different interactive frameworks in [147] and [148] .

Recall from section 2.3, that successful modes of interaction in the literature work closely with the underlying segmentation algorithm. The method here uses run-time interactions to update the posterior model. After initialising the tool the user can identify points on the region boundary that the displayed contour does not pass through. The software calculates the angle θ'_j and radius r'_j corresponding to this point and defines a *noise-free observations* by $\mathcal{N}(r'_j, 0)$ corresponding to angle θ_j. The observation and angle θ'_j complement the observation model above and the MAP solution is re-calculated, which passes

through the user-defined boundary point. The new solution is displayed in real time and any number of similar interactions can be performed to further refine the model. Figure 8.8 demonstrates the use of interactions as noise-free observations.



(a) (b) (c) (d)

Figure 8.8: Using the GP SSM tool for interactive contouring of the same MS lesion image as in figure 8.6 (a). *Top row:* (a) The first contour (green) displayed after the user has estimated the region centre $\mathbf{x}'_c$ (red). (b-d) Post editing by successively identifying one boundary point (b) followed by a second (c) and third (d) shown in red. *Bottom row:* corresponding radial time series in the zero-mean field, where black lines show the MAP solution, grey points show noisy observations $\hat{g} \pm \sigma^g$ from the radial profile model and '⊠' are polar representations of user-identified boundary points used as noise-free observations.

It is interesting to consider an equivalent mode of interaction in the Langevin framework. Such a scheme would require updating the drift and diffusion functions so that the refined model is constrained to pass through a point $\{r, \theta\}$. Changes could only update the transition densities $\Pr(r_{i+1}\theta)|r_i)$ and so the analogy breaks down. An interaction can only assign a high probability $\Pr(r(\theta))$, which not conditional on the previous point. Even if a single transition density $\Pr(r_{i+1}\theta)|r_i)$ could be somehow prescribed, the Markovian nature of the model means that the information would only be used if the radial time series passed through the corresponding radius r_i. Moreover, the constraint would be effective *every* time the series passed through radius r_i, not only at the desired point $\{r_i, \theta_i\}$, i.e. at the angle corresponding to the interaction.

8.6 Data and Performance Evaluation

This section describes how we evaluate the SSM segmentation frameworks. We select test images and perform segmentation with both the RACM and the interactive, tools based on generative SSMs. In all cases we seek to evaluate the power of the shape prior in the corresponding framework. Section 8.6.1 introduces test images common to all following experiments. Section 8.6.2 states what figures of merit

we use, and explains their various roles in measuring accuracy, as well as useability and variability of an interactive tool. Section 8.6.3 explains how we create reference tools that without learned shape knowledge, and how we test for differences in performance between these and the full SSMs.

8.6.1 Test images

To allow us to make more reliable measurement of accuracy, we create synthetic images wherein the true region boundary is known, and is a contour chosen from the set of MS lesion or liver tumour ground truth. To isolate the benefits of the shape models it is desirable to use synthetic images that alleviate the problem of boundary ambiguity. However, we also need to test the segmentation tools in realistic data, for the conclusions to be of practical use. As a compromise we create synthetic images of low SNR and, in the case of MS lesions, repeat experiments on real MRI images.

The foreground and background in synthetic images have Gaussian histograms with mean grey-levels 187 and 210 respectively, and standard deviations 12.5. These statistics are equivalent to a SNR of 1.84, within the range seen for tumour and lesion imaging applications. We also smooth the synthetic images with a 3×3 pixel averaging kernel. Figure 8.9 shows the complete set of test images.

Starting with 241 star-shaped training contours from manual liver-tumour segmentations [29] and 1307 from MS lesions. We remove the three test contours from each set and train Langevin and GP SSMs on the remaining contours.

8.6.2 Figures of merit

We evaluate segmentation tools in terms of accuracy, variability and useability. As in section 6.3.1, we choose performance measures and comparison methods to suit the application and the components of a segmentation framework being evaluated. As with the interactive jetstreams of section 6.3, we evaluate the present segmentation frameworks with a combination of spatial similarity measures and quantitative measures of user behaviour.

In the case of the generative segmentation frameworks, similarity measures each have five distinct roles relating to the two stages of initialisation and post editing. First, the similarity between an *initial* contour and the ground truth indicates the accuracy of the probabilistic algorithms in sections 8.4.3 and 8.5.2, in the absence of post editing. Second, the similarity between a *final* contour accepted by the user and the ground truth indicates the accuracy of the process. However, the 'process' here refers to a combination of the probabilistic framework, the modes of interaction and the user's ability to both use the tool and perceive a region. Given the level of user control and the subjectivity of supervised segmentation, accuracy is best suited to comparing two tools used by the same person. Third, the similarity between a final contour and a second contour, accepted by a different operator using the same tool to segment the same ROI, measures the inter-operator variability of that tool. Similarly, the similarity between a final contour and a second contour, accepted by the same operator using the same tool to segment the same ROI, measures the intra-operator variability of that tool. The final role of similarity relates to the useability of a tool. The similarity between a final contour accepted by the user and the initial contour before post editing, provides a measure of the level of post editing necessary.

In evaluating similarity we use mean minimum distance (MMD) [107] and Dice similarity coef-

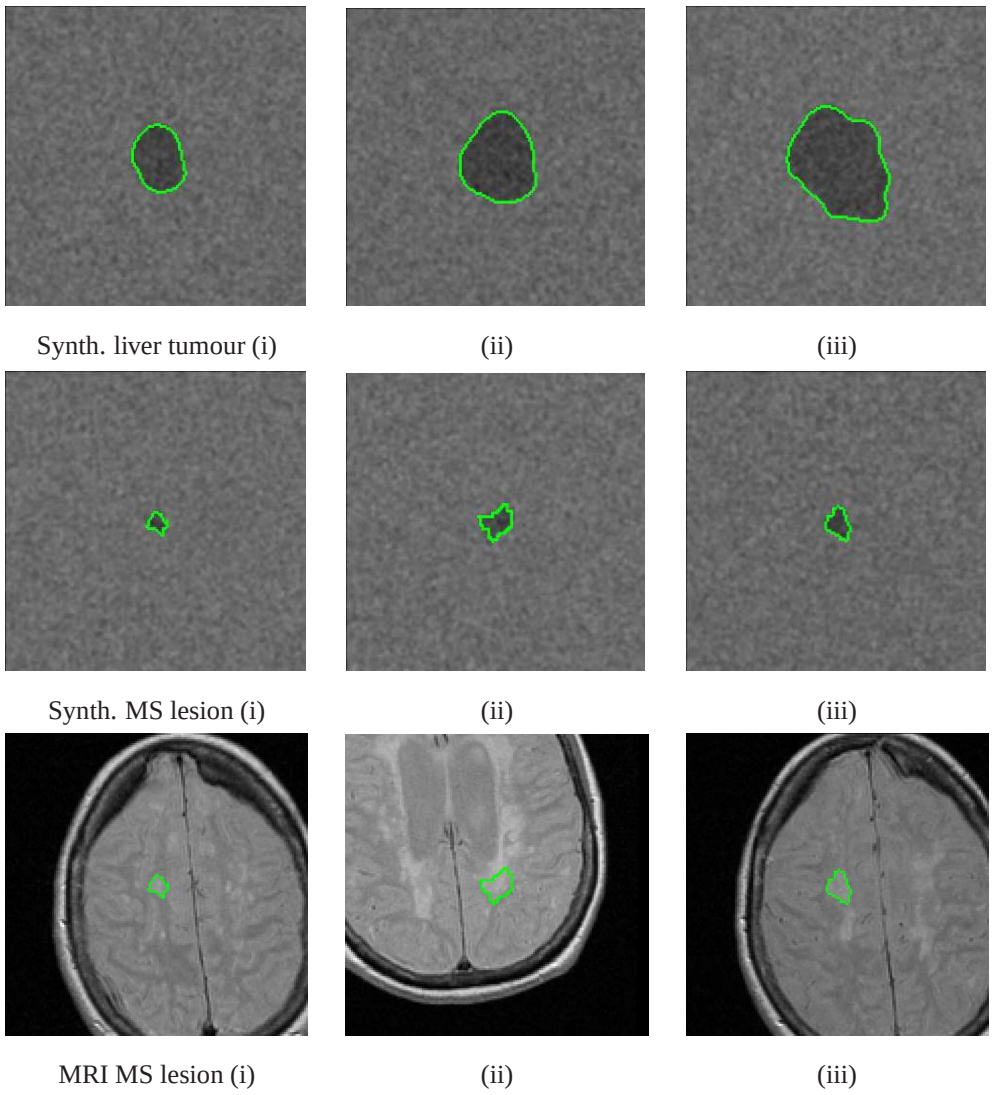


Figure 8.9: ROIs used in experiments, with numbers (i) to (iii) used in subsequent discussions. *Top row:* synthetic liver tumour images. *Middle row:* synthetic MS lesion images. *Bottom row:* MS lesion images from PD weighted MRI.

ficient (DSC) [104]. Following the arguments in section 6.3.1 MMD is appropriate as it is a stable boundary-based dissimilarity measure and the region-based DSC is relevant to the secondary measure of 'lesion load' sought by MS lesion segmentation.

In the case of the generative segmentation frameworks, we also use the Hausdorff distance d_H to measure the similarity between a final contour accepted by the user and the initial contour before post editing [106]. Recall that d_H is a 'maxmin' measure, giving the largest example around the whole of a contour, of the shortest distance from a pixel on that contour to any pixel on the compared contour. This offers meaningful quantification of the amount of post editing performed using methods in sections 8.4.3.1 and 8.5.2.1, as these methods make local corrections where the contour deviates from the true boundary. In the case of shape regularisation we do not implement any post editing procedures, so we just use the MMD and DSC.

In the case of the generative frameworks, which are interactive, we also measure the useability of each tool using two aspects of user behaviour. First, we take the number of boundary points interacted with ('drags' in section 8.4.3.1 or 'noise-free observations' in section 8.5.2.1). The second aspect of user behaviour is the number of initialisations deemed necessary before a contour is edited (or accepted without editing).

8.6.3 Comparison methods

We have simultaneously developed new shape models along with interactive DCM frameworks for their use in segmentation. Direct comparison with another segmentation algorithm in the literature is considered as future work, while we currently design 'proof of concept' experiments to test if the shape models are effective and the segmentation algorithms are useful. We take the same general approach as in [73, 42, 149, 53], which is to test the value of a new *addition* to a segmentation framework (eg shape prior, image prior or interactive mode) by comparing the same framework with and without that addition.

Recall that, in the case of SVM jetstreams, we evaluated the role of the texture classifiers by considering an equivalent tool driven by intensity gradient as a comparison method. In the case of the Langevin and GP frameworks we seek to evaluate the benefits of the global shape priors, which calls upon different comparison methods for the shape regularisation framework (RACM) and generative frameworks. For the shape regularisation framework we duplicate the algorithm without the shape prior by setting $\alpha = 1$ to remove the shape scoring from the energy functional. For the generative frameworks, removing the global shape prior is not trivial. We wish to replace the learned shape information with something that is reasonable, but does not assume any prior knowledge about the global shape of a region. In the case of the Langevin model we choose a stationary distribution $\Pr(r_{i+1}|r_i) = \mathcal{N}(r_i, 1)$ to replace the transition densities. This results in a subtly different tool, where local smoothness is retained but the global drift and diffusion characteristics are removed. In the GP model we would ideally replace the covariance matrix with the $N \times N$ identity matrix $\Sigma(\mathbf{r}, \mathbf{r}) = \mathbf{I}$, equivalent to using the Kronicker delta function as the kernel. However, this leads to numerical issues as near-singular matrices need to be inverted. Instead we use a covariance kernel that approximates the delta function $\varepsilon(r_i, r_i) \sim \delta(\theta_i - \theta_j)$ by setting $\varepsilon(r_i, r_i) = \min[1, (N(\theta_j - \theta_i)^{-1})]$. In both Langevin and GP experiments we refer to their respective reference models as having a *normal prior* as opposed to a *learned prior*.

For a given measure of accuracy, variability or useability, we look for significant differences between results for a framework with learned and normal prior. For this purpose we use both parametric and non-parametric hypothesis tests. First we use the t-test (parametric) as before. We also use the Wilcoxon signed rank test (non-parametric), which is appropriate when the assumption of normally distributed results. The non-parametric test is more conservative, making less assumptions about the data than the t-test, and might alleviate problems associated with interpretation of results from small samples discussed in section 6.4.

8.7 Experiments

This section evaluates the discriminative and generative shape models for segmentation. Section 8.7.1 evaluates the benefits of the Langevin model for shape regularisation in star-shaped region segmentation. As stated above, the regularisation method extends naturally for the generalised model with arc-length parametrisation 7.1 (a) We defer evaluation of the generalised case for future work.

Section 8.7.2 evaluates the interactive generative frameworks used to segment star-shaped regions.

8.7.1 Shape regularisation

Experiments in this section test the use of discriminative Langevin models for shape regularisation in the RACM framework described in section 8.2. We choose not to use the GP model for shape regularisation as discussed in the same section.

First, we test the ability of the regularised RACM to balance information regarding image and shape properties. We noted in section 2.5 that conflicting image and shape information need to be balanced and that greedy algorithms can enter an oscillatory state without convergence. We hypothesise that:

$\mathcal{H}$8.7.1.1: The evolution of a RACM terminates by convergence to zero energy

To test this hypothesis we run the RACM on the test images. For each region we fix the centre-point $\mathbf{x}_c$ at the 'true' centre as described in section 7.1.2 and sample radial profiles of the gradient magnitude as described in section 8.1.1. We estimate $\bar{r}$ from the mean of vector $\hat{\mathbf{g}}$ in the observation model and initialise the RACM with a noisy circle, defined by drawing r_i from $\mathcal{N}(\bar{r}, 2.5)$ for $i = \{0, \ldots, N-1\}$. As described section 8.2, the evolving RACM terminates when the total energy is zero *or* when reaching the chosen maximum number of iterations. We set this maximum number to 500. Table 8.2 shows the number of iterations for convergence in each of the test images.

α	Synthetic						MRI		
	liver tumour			MS lesion			MS lesion		
	(i)	(ii)	(iii)	(i)	(ii)	(iii)	(i)	(ii)	(iii)
0.7	311	458	406	73	65	66	111	214	113
0.8	308	166	364	71	76	99	107	176	58
1	115	123	120	32	46	36	52	41	46

Table 8.2: Number of iterations for convergence of the RACM algorithm

In all cases the algorithm terminates at zero energy (< 500 iterations) allowing us to accept hypothesis $\mathcal{H}$8.7.1.1.

Next, we test the accuracy of the regularised RACM. We hypothesise that:

$\mathcal{H}$8.7.1.2: the accuracy of a simple deformable contour model is improved by the use of Langevin shape priors.

To test this hypothesis we perform quantitative experiments using the RACM algorithm without interactivity. We initialise the RACM in the test images in figure 8.9. Upon convergence to zero energy

we evaluate RACM accuracy by calculating the mean minimum distance (MMD), Dice similarity coefficient (DSC) and Hausdorff distance d_H with respect to the ground truth contour. We repeat for each of the test images both with and without shape regularisation, where shape regularisation is included by setting $\alpha < 1$ in equation 8.10, and excluded by $\alpha = 1$. Tables 8.3 to 8.5 shows the results for images (i) to (iii), in each of three image types as in figure 8.9, along with the overall mean. In all cases the shape regularisation ($\alpha = 0.7$ or 0.8) gives an increase in accuracy, indicated by lower mean MMD or d_H, or higher mean DSC.

Next we test for significant differences between the accuracy of the RACM with and without shape regularisation using a one-tailed paired-samples t-test. The p-values in tables 8.3 to 8.5 indicate the significance of differences in accuracy between the RACM without shape prior and the cases when $\alpha = 0.7$ and $\alpha = 0.8$. Superscript '+' denotes an increase in accuracy (no results give a decrease in accuracy), when using shape priors (non-zero α). Bold values indicate where differences are significant with a confidence interval of 95%. Results are given separately for each image segmented.

α	Synthetic						MRI			mean	p-value
	liver tumour			MS lesion			MS lesion				
	(i)	(ii)	(iii)	(i)	(ii)	(iii)	(i)	(ii)	(iii)		
0.7	0.53	1.01	1.60	0.72	0.92	0.89	0.87	1.72	0.90	1.018±0.391	0.077^{+}
0.8	0.57	1.00	1.68	0.73	0.61	0.78	1.08	1.91	1.02	1.042±0.466	$\mathbf{0.028}^{+}$
1	0.65	1.24	1.89	0.72	0.69	0.76	1.35	2.06	0.90	1.140±0.534	N/A

Table 8.3: Effect of learned shape regularisation on segmentation accuracy in terms of mean minimum distance (MMD).

α	Synthetic						MRI			mean	p-value
	liver tumour			MS lesion			MS lesion				
	(i)	(ii)	(iii)	(i)	(ii)	(iii)	(i)	(ii)	(iii)		
0.7	0.97	0.95	0.94	0.79	0.83	0.85	0.74	0.71	0.84	0.847±0.092	0.270^{+}
0.8	0.96	0.95	0.94	0.78	0.87	0.86	0.71	0.69	0.82	0.842±0.101	0.407^{+}
1	0.96	0.93	0.93	0.79	0.88	0.86	0.71	0.67	0.84	0.841±0.101	N/A

Table 8.4: Effect of learned shape regularisation on segmentation accuracy in terms of Dice Similarity Coefficient (DSC)

Tables 8.3 to 8.5 show that, for the boundary-based accuracy measures of MMD and d_H, regularisation gives significant improvement for one or both of the tested shape prior weightings ($\alpha = 0.7$ or 0.8). In terms of the region-based accuracy measure (DSC) the benefit of shape regularisation is not significant.

In conclusion, we accept hypothesis $\mathcal{H}$8.7.1.2 for boundary-based accuracy, but not region-based, where the improvement is apparent but not significant.

α	Synthetic						MRI			mean	p-value
	liver tumour			MS lesion			MS lesion				
	(i)	(ii)	(iii)	(i)	(ii)	(iii)	(i)	(ii)	(iii)		
0.7	1.41	13.0	15.62	1.41	3.0	3.16	7.62	7.81	2.24	6.141±5.251	**0.023**$^+$
0.8	1.41	15.65	16.64	2.83	2.0	2.0	8.54	8.06	2.24	6.597±6.032	**0.014**$^+$
1	2.24	24.17	18.03	5.00	2.24	4.47	10.0	8.94	4.24	8.814±7.607	N/A

Table 8.5: Effect of learned shape regularisation on segmentation accuracy in terms of Hausdorff distance (d_H)

8.7.2 Interactive generative frameworks

Experiments in this section test the practical value of the interactive segmentation frameworks of sections 8.4 and 8.5, and the benefits of the dynamical shape priors to these frameworks. We design an experimental protocol that tests the tools in various scenarios and calculate statistics of each tool's performance in user trials.

Following the findings above, we set $\alpha = 0.75$ in both Langevin and GP frameworks. The number of shape-wise particles M in the shape-wise importance sampling common to both frameworks must balance the benefits of this part of the algorithm with a linear increase in computation time. In practice we find that reducing M to 5 allows real-time segmentation without compromising accuracy.

We ask 10 volunteers to use both segmentation frameworks in the following experiments. Each volunteer uses a tool to segment the same randomised sequence of images from the set in figure 8.9. The sequence includes each region twice for segmentation by the tool with normal and learned priors. In addition, one region is included a further two times to allow repeated segmentation by each prior type. For this purpose we choose region (iii), from each image type (synthetic liver tumour, synthetic MS lesion and MRI MS lesion). The resulting sequence comprises 24 segmentations, which a user repeats using Langevin and GP frameworks.

Both Langevin and GP frameworks involve an initialisation step followed by interactive post editing. In each case the user can repeat the initialisation any number of times. After initialisation, the user either accepts a contour without post editing, or performs any amount of interactions. These interactions are either 'drags' in the case of the Langevin tool (figure 8.6) or noise-free observations (mouse clicks on the boundary) in the case of the GP tool (figure 8.8).

Before performing the experiment, each volunteer practises by segmenting 3 regions (not from the set in figure8.9) using each tool with learned and normal priors. Whilst using a given tool (Langevin or GP) a user is not aware that it occurs in two different modes (learned and normal prior).

The remainder of this section presents various statistical analyses that test certain hypotheses regarding the benefits of the shape priors to the accuracy, variability and useability of each tool.

8.7.2.1 Accuracy

First, we hypothesise that:

$\mathcal{H}$8.7.2.1: the accuracy of each interactive framework is increased by the learned global shape priors.

To test hypothesis $\mathcal{H}$8.7.2.1 we measure the dissimilarity, in terms of MMD and DSC, between segmentation results and ground truth and take the mean dissimilarity over all regions. Results for each user are given in appendix tables 10.1 (for synthetic liver tumours), 10.2 (for synthetic MS lesions) and 10.3 (for MRI MS lesions) along with the overall mean MMD and DSC for each tool. For synthetic liver tumours (table 10.1) the mean accuracy is the same or superior for tools with learned shape priors, with the exception of the accuracy of the Langevin tool as measured by MMD, which shows a small (0.022 pixel units) increase in MMD when the prior is used. For synthetic MS lesions (table 10.2) the mean accuracy is consistently superior for tools with learned shape priors. For MRI MS lesions (table 10.3) the mean accuracy is the same or superior for tools with learned shape priors, with the exception of the accuracy of the Langevin tool as measured by DSC.

To look for significant differences between the accuracy of the tools used with learned and normal priors we use a paired-samples t-test and a non-parametric (Wilcoxon signed rank) test. Table 8.6 shows the p-values revealed by both tests, when similarity is measured by mean minimum distance (MMD) and Dice similarity coefficient (DSC). The p-values in table 8.6 indicate the significance of the difference in accuracy resulting from learned shape priors. Superscripts '+' and '-' denote an increase and decrease in accuracy respectively, when using learned shape priors. Bold values indicate where differences are significant with a confidence interval of 95%. No results give a significant reduction in accuracy. Results are given separately for synthetic liver tumour (LT) and multiple sclerosis (MS) regions as well as the real MRI images of MS lesions.

Model	Measure	T-test			Wilcoxon		
		synth. LT	synth. MS	MRI MS	synth. LT	synth. MS	MRI MS
Lan	MMD	0.287^{-}	0.485^{+}	$\mathbf{0.050}^{+}$	0.361^{-}	0.480^{+}	0.057^{+}
	DSC	0.476^{-}	0.815^{+}	0.340^{-}	0.439^{-}	0.193^{+}	0.228^{-}
GP	MMD	$< \mathbf{0.001}^{+}$	0.250^{+}	0.064^{+}	$\mathbf{00.003}^{+}$	0.288^{+}	0.057^{+}
	DSC	$< \mathbf{0.001}^{+}$	0.194^{+}	$\mathbf{0.006}^{+}$	$\mathbf{0.003}^{+}$	0.288^{+}	$\mathbf{0.004}^{+}$

Table 8.6: p-values indicating significance of the effect of learned shape priors on the accuracy of interactive segmentation

Table 8.6 leads to the following observations:

- According to a parametric test, the accuracy of the Langevin framework, as measured by MMD, is significantly increased by the learned shape prior when segmenting MRI MS lesions.

- The accuracy of both tools is consistently increased when segmenting MS lesions in synthetic images, but this increase is not significant.

- According to both parametric and non-parametric tests, the accuracy of the GP framework, as

measured by MMD *and* DSC, is significantly increased by the learned shape prior when segmenting synthetic liver tumours, and that measured by MMD alone is significantly increased by the learned shape prior when segmenting MRI MS lesions.

In conclusion, we accept hypothesis $\mathcal{H}$8.7.2.1 in the Langevin case for MRI MS lesions and in the GP case for all regions except for synthetic MS lesions. The apparent (but insignificant) reduction in accuracy for the Langevin tool segmenting synthetic liver tumours could caused by a reduction in post editing. The tool with shape priors may give a reasonable contour, which subconsciously influences the user to accept the result with insufficient post editing. Indeed, this post editing involved on average 10.520 boundary interactions for the tool with shape priors and 16.075 without.

8.7.2.2 Inter-operator variability

Next, we hypothesise that:

$\mathcal{H}$8.7.2.2: inter-operator variability of each interactive framework is reduced by the global shape priors.

To test hypothesis $\mathcal{H}$8.7.2.2 we measure the dissimilarity, in terms of MMD and DSC between the segmentation by two different users, of a single ROI. For each ROI we take the mean over all 45 distinct pairs of users. Because each region can be perceived differently by each user we give results for each region separately. Table 10.4 gives the results separately for segmentation of synthetic (liver tumour and MS lesion) and MRI (MS lesion) images. In the Langevin framework, learned shape priors reduce the inter-operator variability measured by MMD and DSC in 5 out of the 9 images, namely synthetic liver tumours (i) and (iii), synthetic MS lesions (i) and (iii) and MRI MS lesion (iii), as well as synthetic liver tumour (ii) in terms of MMD alone. In the GP framework, learned shape priors reduce the inter-operator variability measured by MMD and DSC in 3 out of the 9 images, namely synthetic liver tumour (i), synthetic MS lesion (ii) and MRI MS lesion (ii), as well as MRI MS lesion (i) in terms of MMD alone.

To look for significant differences between the inter-operator variability of the tools used with learned and normal priors we use a paired-samples t-test and a non-parametric (Wilcoxon signed rank) test. Tables 8.7 and 8.8 show the p-values revealed by parametric and non-parametric tests respectively, when similarity is measured by mean minimum distance (MMD) and Dice similarity coefficient (DSC). The p-values in tables 8.7 and 8.8 indicate the significance of the difference in inter-operator variability resulting from learned shape priors. Superscripts '+' and '-' denote a reduction and increase in inter-operator variability respectively, when using learned shape priors. Bold values indicate where differences are significant with a confidence interval of 95%. Results are given separately for each image segmented. Tables 8.7 and 8.8 lead to the following observations:

- For some images, the reduction in inter-operator variability is significant in terms of one or both measures (MMD/DSC) according to one or both tests (parametric/non-parametric). In the case of the Langevin framework this is true for synthetic liver tumours (i) and (ii), synthetic MS lesions (i) and (iii) and MRI MS lesion (iii). In the case of the GP tool this is true for synthetic liver tumour (i), synthetic MS lesion (ii) and MRI MS lesion (ii).

Mod.	Meas.	Synthetic						MRI		
		liver tumour			MS lesion			MS lesion		
		(i)	(ii)	(iii)	(i)	(ii)	(iii)	(i)	(ii)	(iii)
Lan	MMD	$<$ **.001**$^{+}$	0.338^{+}	**0.006**$^{+}$	**0.008**$^{+}$	**0.047**$^{-}$	0.278^{+}	0.359^{-}	**0.011**$^{-}$	**0.008**$^{+}$
	DSC	0.159^{+}	**0.019**$^{-}$	$<$ **.001**$^{+}$	**0.002**$^{+}$	0.208^{-}	$<$ **.001**$^{+}$	**0.003**$^{-}$	**0.011**$^{-}$	0.055^{+}
GP	MMD	0.087^{+}	0.321^{-}	**0.018**$^{-}$	0.245^{-}	**0.010**$^{+}$	**0.032**$^{-}$	0.260^{+}	0.323^{+}	0.098^{-}
	DSC	**0.046**$^{+}$	0.462^{-}	0.056^{-}	0.305^{-}	**0.009**$^{+}$	**0.002**$^{-}$	0.061^{-}	**0.030**$^{+}$	0.490^{-}

Table 8.7: p-values from T-test indicating significance of the effect of learned shape priors on the inter-operator variability of interactive segmentation

Mod.	Meas.	Synthetic						MRI		
		liver tumour			MS lesion			MS lesion		
		(i)	(ii)	(iii)	(i)	(ii)	(iii)	(i)	(ii0	(iii)
Lan	MMD	$<$**.001**$^{+}$	0.357^{+}	**0.008**$^{+}$	**0.019**$^{+}$	0.141^{-}	0.328^{+}	0.400^{-}	**0.005**$^{-}$	**0.012**$^{+}$
	DSC	0.230^{+}	**0.025**$^{-}$	$<$ **.001**$^{+}$	**0.015**$^{+}$	0.255^{-}	**0.001**$^{+}$	**0.006**$^{-}$	**0.012**$^{-}$	0.082^{+}
GP	MMD	0.136^{+}	0.444^{-}	0.136^{-}	0.489^{-}	**0.021**$^{+}$	0.053^{-}	0.154^{+}	0.173^{+}	0.123^{-}
	DSC	0.063^{+}	0.431^{-}	0.084^{-}	0.484^{-}	**0.015**$^{+}$	**0.003**$^{-}$	0.077^{-}	**0.039**$^{+}$	0.480^{-}

Table 8.8: p-values from Wilcoxon signed rank test indicating significance of the effect of learned shape priors on the inter-operator variability of interactive segmentation

- For some of the images revealing an *increase* in inter-operator variability, this increase is significant in terms of one or both measures (MMD/DSC) according to one or both tests (parametric/non-parametric). In the case of the Langevin framework this is true for synthetic liver tumour (ii), synthetic MS lesion (ii) and MRI MS lesions (i) and (ii).

- To summarise results for the Langevin framework, the parametric tests reveal 7 cases where the inter-operator variability is significantly reduced and 5 cases where it is significantly increased. These numbers become 4 and 0 in the case of non-parametric tests.

- To summarise results for the GP framework, the parametric tests reveal 4 cases where the inter-operator variability is significantly reduced and 3 cases where it significantly increased. These numbers become 3 and 1 in the case of non-parametric tests.

In conclusion, we can not accept hypothesis $\mathcal{H}$8.7.2.2 due to the appreciable amount of significant negative results. These results could be explained by the high levels of user control offered by both post editing methods. More user control allows for different styles of post editing between users, which can override the benefits of the shape priors to the variability of results.

8.7.2.3 Intra-operator variability

Next, we hypothesise that:

> $\mathcal{H}$8.7.1.3: intra-operator variability of each interactive framework is reduced by the global shape priors.

We measure the similarity, in terms of MMD and DSC between two segmentations of a region (iii) by the same user at different times. The full results are given in appendix tables 10.5 (for synthetic liver tumours), 10.6 (for synthetic MS lesions) and 10.7 (for MRI MS lesions) along with the overall mean MMD and DSC for each tool.

Results for each user are given in appendix tables 10.1 (for synthetic liver tumours), 10.2 (for synthetic MS lesions) and 10.3 (for MRI MS lesions) along with the overall mean MMD and DSC for each tool. For synthetic liver tumours the mean intra-operator variability is the same or superior for tools with learned shape priors, with the exception of the intra-operator variability of the Langevin tool as measured by MMD. For synthetic MS lesions the mean intra-operator variability is the same or superior for tools with learned shape priors, with the exception of the intra-operator variability of the Langevin tool as measured by MMD and the GP tool as measured by DSC. For MRI MS lesions the mean intra-operator variability is consistently superior for tools with learned shape priors.

To test hypothesis $\mathcal{H}$8.7.1.3 we look for significant differences between the intra-operator variability of the tools used with learned and normal priors using a paired-samples t-test and a non-parametric (Wilcoxon signed rank) test. Table 8.9 shows the p-values revealed by both tests, for each image type and when similarity is measured mean minimum distance (MMD) and Dice similarity coefficient (DSC). Superscripts '+' and '-' denote a reduction and increase in intra-operator variability respectively, when using learned shape priors. None of the differences are significant with a confidence interval of 95%.

Model	Measure	T-test			Wilcoxon		
		synth. LT	synth. MS	MRI MS	synth. LT	synth. MS	MRI MS
Lan	MMD	0.063^{-}	0.279^{-}	0.128^{+}	0.121^{-}	0.430^{-}	0.143^{+}
	DSC	0.279^{+}	0.500^{-}	0.405^{+}	0.322^{+}	0.399^{-}	0.288^{+}
GP	MMD	0.184^{+}	0.377^{-}	0.458^{+}	0.254^{+}	0.323^{-}	0.400^{+}
	DSC	0.323^{+}	0.268^{-}	0.494^{-}	0.480^{+}	0.305^{-}	0.480^{-}

Table 8.9: p-values indicating significance of the effect of learned shape priors on the intra-operator variability of interactive segmentation

Table 8.9 suggests that the intra-operator variability of both Langevin and GP frameworks is reduced in all cases except for synthetic MS lesion segmentation, which involves the smallest regions.

In conclusion, we can not accept hypothesis $\mathcal{H}$8.7.1.3 due to the lack of significant difference between intra-operator variability of the tools with learned and normal priors. These results could be explained by the high levels of user control as in the case of $\mathcal{H}$8.7.2.2.

8.7.2.4 Useability

Next, we hypothesise that:

> $\mathcal{H}$8.7.2.4: the number of initialisations necessary in each framework is reduced by the global shape priors.

We test hypothesis $\mathcal{H}$8.7.2.4 in two ways. First, we count how many times number of initialisations $N_{\text{init}} = 1$. In these special cases a user accepts or edits a contour after a single initialisation. We compare this number for each framework with and without learned shape priors. For the Langevin framework, $N_{\text{init}} = 1$ 72 times with learned prior and 54 with normal prior. For the GP framework, $N_{\text{init}} = 1$ 103 times with learned prior and 95 times with normal prior. In both cases the learned shape prior leads to an increase in the number of once-only initialisations, where the difference is more apparent in the Langevin case.

Second, we count the number of initialisations N_{init} that a user invokes before accepting or editing a contour. We take the mean over all regions and compare for tools with normal and learned prior, reporting the difference for each user. Table 10.8 shows the results for each user along with overall mean and standard deviation. We also look for significant differences between the number of initialisations of the tools used with learned and normal priors using a paired-samples t-test and a non-parametric (Wilcoxon signed rank) test. The p-values in table 8.10 indicate the significance of the difference in the number of initialisations resulting from learned shape priors. Superscripts '+' denote a reduction in the number of initialisations, when using learned shape priors. Bold values indicate where differences are significant with a confidence interval of 95%. All results give a significant reduction in the number of initialisations. Table 8.10 reveals that, for both Langevin and GP frameworks, the learned shape

Model	T-test	Wilcoxon
Lan	$\mathbf{0.017}^{+}$	$\mathbf{0.008}^{+}$
GP	$\mathbf{0.039}^{+}$	$\mathbf{0.046}^{+}$

Table 8.10: p-values indicating significance of the effect of learned shape priors on the number of initialisations

priors lead to a significant reduction in the number of contour initialisations necessary, according to both parametric and non-parametric tests.

In conclusion, we accept hypothesis $\mathcal{H}$8.7.2.4 for both segmentation frameworks and all image types.

Next, we hypothesise that:

> $\mathcal{H}$8.7.2.5: The level of post editing necessary in each framework is reduced by the global shape priors

We test hypothesis $\mathcal{H}$8.7.2.5 in two ways. First, we count how many times the number of boundary point interactions made $N_{\text{int}} = 0$. In these special cases a user accepts a contour after initialisation

without post editing. We compare this number for each framework with and without learned shape priors. For the Langevin framework,this happens 10 times with learned prior and once with normal prior. For the GP framework, this happens 16 times with learned prior and 9 times with normal prior. In both cases the learned shape prior leads to a striking increase in the number of first-time acceptances.

Second, we compare the level of post editing performed on initial contours for each tool used with normal and learned shape priors. We quantify the 'level' of post editing in two ways. First, we count the number of boundary point interactions made N_{int}, being the number of points 'dragged' in the Langevin framework (figure 8.6) and the number of noise-free observations in the GP case (figure 8.8). Second, to account for the subjective nature of the level of post editing deemed 'necessary' by a user, we measure the dissimilarity between initial and final contours using the Hausdorff distance d_H. In each case (N_{int} and d_H) we take the mean over all regions and compare for tools with normal and learned prior. The full results are given in appendix tables 10.9, 10.10 and 10.11. We also look for significant differences between level of post editing of the tools used with learned and normal priors using a paired-samples t-test and a non-parametric (Wilcoxon) test. The p-values in table 8.11 indicate the significance of the difference in the level of post editing resulting from learned shape priors. Superscript '+' denotes a reduction in the level of post editing (no results give an increase in the level of post editing), when using learned shape priors. Bold values indicate where reductions are significant with a confidence interval of 95%. Results are given separately for synthetic liver tumour (LT) and multiple sclerosis (MS) regions as well as the real MRI images of MS lesions.

Model	Measure	T-test			Wilcoxon		
		synth. LT	synth. MS	MRI MS	synth. LT	synth. MS	MRI MS
Lan	N_{int}	$\mathbf{0.002}^{+}$	0.334^{+}	0.157^{+}	$\mathbf{0.004}^{+}$	0.323^{+}	0.143^{+}
	d_H	$\mathbf{0.019}^{+}$	0.173^{+}	0.418^{+}	$\mathbf{0.024}^{+}$	0.143^{+}	0.323^{+}
GP	N_{int}	$< \mathbf{0.001}^{+}$	$\mathbf{0.010}^{+}$	$\mathbf{0.001}^{+}$	$< \mathbf{0.001}^{+}$	$\mathbf{0.013}^{+}$	$\mathbf{0.004}^{+}$
	d_H	$\mathbf{0.021}^{+}$	0.185^{+}	0.299^{+}	$\mathbf{0.021}^{+}$	0.180^{+}	0.288^{+}

Table 8.11: p-values indicating significance of the effect of learned shape priors on the level of post editing

Table 8.11 leads to the following observations:

- The level of post editing is significantly reduced by learned shape priors when using Langevin or GP frameworks to segmenting synthetic liver tumours. This is true regardless of the statistical test or method of measuring the level of interactivity.

- In terms of the number of interactions N_{int}, the level of post editing is significantly reduced by learned shape priors when using the GP model to segment any of the region/image types. This is true regardless of the statistical test.

We accept hypothesis $\mathcal{H}$8.7.2.5 for both segmentation frameworks and all image types.

8.8 Conclusions and Future Work

This chapter has shown that two new dynamical shape models, namely Langevin and Gaussian process SSMs, can be used in region segmentation. We constructed a simple ACM algorithm for a radial contour parametrisation (RACM) and introduced the use of dynamical shape models for regularisation by incorporating discriminative models in an energy functional. We demonstrated this with the Langevin SSM.

We also presented a generalised framework for interactive segmentation using generative shape models. This framework uses samples drawn from a prior distribution over shapes, along with appropriate observation models from image and interactions, in a Bayesian optimisation scheme. We demonstrated for the case of generative Langevin and GP SSMs, for which we presented methods of generating model shapes, incorporating image observations and, in the case of GP SSMs, make efficient use of run-time information from user interactions. The incorporation of observations to condition the shape priors is a major contribution.

We performed experiments to isolate the learned shape information and test its benefit to the rest of a segmentation framework. These experiments reveal that

- a simple deformable contour model (RACM) that combines Langevin shape regularisation with a stochastic deformation mechanism converges to a stable solution in a range of images,
- Langevin shape regularisation improves the accuracy of segmentation by the RACM,
- GP shape regularisation is not practical for run-time segmentation but might benefit reconstruction or registration tasks,
- learned shape priors generally improve the accuracy of interactive segmentation tools based on Langevin and GP SSMs,
- prior knowledge of shape does not reduce segmentation variability in a framework that gives ultimate control to the user, which echoes the conclusion drawn from chapter 6, and
- the demand on the user of an interactive framework is reduced when exploiting learned shape priors.

The experiments above represent a 'proof of concept' for the new shape models and their respective segmentation algorithms. The next step would be to compare the methods with the closest competitor in the literature. It would be interesting, for example, to compare the power of the shape priors, with those encoded by a PDM, built without explicit point correspondence as in Berks *et al.* [130]. We noted in section 3.1.1, that using the method of Berks *et al.* for supervised segmentation would first require the incorporation of the simulation algorithm in a segmentation framework. The generalised framework above could incoprorate the method as a generative mechanism, i.e. (4) in table 8.1.

This chapter developed and tested SSMs for star-shaped regions only. In the case of shape regularisation, the extension to the generalised contour parametrisation is straightforward, although the polar image model used here needs to be replaced. Future work will evaluate the SSMs for generalised shape

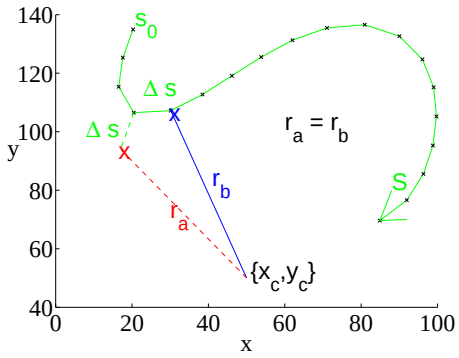


Figure 8.10: Ambiguity for the contour parametrisation of radii $\mathbf{r}$ vs. arc-length $\mathbf{s}$.

regularisation and for different segmentation methods such as classical snakes and level sets. A framework that parametrises a contour as a Fourier decomposition could also benefit from this type of shape regularisation following the findings in the previous chapter (section 7.5.2). In the case of the generative frameworks, extension to non star-shaped regions requires more changes to the algorithm. The arc-length parametrisation $r(s)$ introduces an ambiguity problem. For any starting point s_0, there is no 1:1 mapping between a point $\{r_i, s_i\}$ and a point $\{x, y\}$ in the image frame as shown in figure 8.10. To overcome this problem, future work will replace the radial time series with an *angular time series* $\{\phi, \mathbf{s}\}$, where $\{\phi = \{\phi_0, \phi_1, \ldots, \phi_{N-1}\}$ are angles with respect to the horizontal in the image frame and $\mathbf{s} = \{s_0, \ldots, s_i, \ldots s_{N-1}\}$ are arc-length increments. By using this representation with the Langevin model above we could create an open-contour model for interactive boundary tracking with shape priors. However, the models would lose rotation invariance.

Finally, the models in this chapter would extend to 3-dimensions. One approach would be to propogate the 2-dimensional model from one slice to the next, until the whole of a 3-dimensional volume is segmented. Another approach would be to reduce a 3-dimensional shape to a 1-dimensional representation as in [278].

Chapter 9

Conclusions and Future Work

This project has tackled the problem of segmenting difficult regions of interest in medical images, where interactive, 2-dimensional contouring is in common practice but where these lack automation and the applications offer little prior knowledge. We introduced image models and modes of interaction to a boundary tracking framework designed for lesion contouring, and novel statistical shape models designed to introduce global shape priors to supervised contouring. We demonstrated the benefits of new interactive frameworks, as well as the specific roles of the image and shape models, in terms of segmentation quality, variability and the useability of a tool. Section 9.1 presents a condensed form of the findings and contributions made by this project and section 9.2 suggests future research motivated by these findings.

9.1 Conclusions

This section concludes the thesis by summarising the findings and contributions resulting from the work herein.

9.1.1 Key findings

This project has made the following key observations:

1. SVM texture classification leads to better edge detection than gradient filtering, without calculating explicit texture features.

2. The combination of SVM texture models and jetstream interactions benefits segmentation in terms of accuracy and user demand.

3. In a framework that gives ultimate control to the user, prior knowledge from an SVM texture model does not reduce segmentation variability.

4. Two new SSMs based on nonlinear dynamics capture global shape information with high discrimination power, and can be used in applications without correspondence points or other high-level shape similarity, such as tumours and lesions.

5. A simple active contour model benefits from shape priors embedded in the SSMs, in terms of accuracy.

6. A generalised interactive segmentation framework can use different generative shape models as the basis of a probabilistic optimisation scheme.

7. Two dynamical shape models (Langevin and GP SSMs) work well in the generalised segmentation framework, and can incorporate observation models in an efficient and novel manner.

8. Prior knowledge of shape, embedded in the generative models, benefits the interactive segmentation frameworks in terms of accuracy and useability.

9. In a framework that gives ultimate control to the user, prior knowledge from an SVM texture model or time series shape model does not reduce segmentation variability.

9.1.2 Other contributions

We have made the following contributions to the various fields encompassed by this multidisciplinary project:

1. Motivated the use of incremental learning in MS lesion segmentation, by training 'small' SVMs on local data.

2. Presented novel modes of interaction for a boundary tracking framework, including two methods of 'loop closing'.

3. Highlighted the importance of user preference when designing modes of interaction.

4. Motivated new research into the use of time series modelling for shape modelling, which draws from the rich nonlinear dynamics literature beyond the autoregressive and Markov random field models.

5. Introduced techniques from time series analysis, for training SSMs, scoring unseen shapes, simulating model contours and constraining simulations in the light of observations.

6. Presented a novel method of 'data assimilation' for Langevin modelling, by combining simulation techniques with theory from the tracking literature.

7. Stimulated the wider field of medical image analysis by prompting the future avenues of research detailed below.

9.2 Future Work

This project has motivated solid avenues of future work, including novel extensions for which we have proposed clear starting points. We start by listing the future work, roughly in the order according to the thesis, and finish by prioritising the three most pressing matters and recap our suggested approaches.

1. Combining both featureless texture models and correspondence-free shape priors in a unified framework for supervised segmentation of variable shapes with ambiguous boundaries.

2. Using either of the constrained jetstream algorithms, which terminate at a fixed point, as a fast and accurate post editing tool for replacing partial boundary sections in any deformable contour model.

3. Extending Langevin and GP models to other applications than MS lesion and liver tumour contouring, and exploring novel covariance kernels in the GP case.

4. Using discriminative Langevin and GP SSMs for regularisation in other deformable contour frameworks such as level sets

5. Using discriminative Langevin and GP SSMs for classifying MS lesions in terms of 'Lassman types'.

6. Using discriminative Langevin and GP SSMs for regularisation in image registration tasks, where source and target images from different time points are known to contain tumours or lesions

7. Using discriminative Langevin and GP SSMs for regularisation in image reconstruction, where the imaging object is known to contain a tumour or lesion and its rough location is known.

8. Using the generative SSM tools in other applications, such as tumour segmentation in 2-dimensional ultrasound images that have very low SNR.

9. Extending the generative SSM tools for 3-dimensional segmentation by contour stacking, whereby the third dimension could comprise a new variable for orthogonal time series modelling and the observation model could use information propagated through image slices.

10. Extending the generative Langevin model for interactive, open-contour boundary tracking that generalises to non star-shaped regions.

The three prioritised tasks for future work relate to the time series shape models, which reflects their novelty. First, having validated the role of the shape priors in improving the accuracy and useability of a segmentation framework, the resulting tool should be compared with others from the contemporary literature. In particular, it would be of great value to assess the advantages and disadvantages of the frameworks over the use of point distribution models for pathological regions of interest, which resorts to arbitrarily assigning points of correspondence, as in [130]. In one case the arbitrary PDM could be built into the generic framework presented in section 8.3. In another case the arbitrary PDM could be built into an active appearance model [279].

The second priority extends the generative models for non-star shaped regions. One approach is to generate series according to the *angular time series* $\{\phi, \mathbf{s}\}$ as suggested in section 8.8, which could be realised for both Langevin and GP models. Another approach in the Langevin case is inspired by the work of Jafari *et al.* [224], who describe microscopic surfaces with Langevin models. In these models the independent variable is replaced by a 2-dimensional field. For region boundaries the analogous scheme is to maintain radius r as the state variable but model the behaviour of $r(x, y)$ rather than $r(\theta)$.

The third priority generalises the models to 3 dimensions, for which section 8.8 suggested two distinct approaches. In the first case, the 2-dimensional models can be repeated throughout successive slices of a 3-dimensional image, which is in line with both the anisotropy of tomographic images and the natural way in which humans interact with 2-dimensional models as discussed previously. In addition this approach would naturally allow information to propagate through tomographic slices by updating the radial profile model to constrain subsequent hypotheses. The second approach would extend the models themselves to describe 3-dimensional shape. This proposed extension is based on the re-parametrisation of a 3-dimensional surface into a 1-d signature resembling the radial time series. Such a parametrisation is realised by the 'spiral' transform, as used in [278]. In this scheme a radial vector originates from the centre of a volume of interest, and traces a path in spherical polar coordinates, from the 'North pole' to the 'South pole'. If the surface were a perfect sphere with radial vector originating from its geometric centre, the resulting time series would be a straight line, analogous to the case of a perfect circle in the 2-dimensional case.

Chapter 10

Appendix

This appendix gives the full set of results from experiments in section 8.7.2. These results were used for statistical analyses in section 8.7.2, which evaluate generative segmentation tools in terms of accuracy (tables 10.1, 10.2 and 10.3), inter-operator variability (table 10.4) and intra-operator variability (tables 10.5, 10.6 and 10.7) as well as useability in terms of number of initialisations (table 10.8) and level of post-editing (tables 10.9, 10.10 and 10.11).

Model	Measure	Prior	User 1	User 2	User 3	User 4	User 5	User 6	User 7	User 8	User 9	User 10	Mean
Lan	MMD	learned	0.901	0.560	0.8566	0.926	0.753	0.745	0.716	0.730	0.729	0.989	0.792 ±0.126
		normal	0.686	0.583	0.965	0.942	0.747	0.893	0.730	0.655	0.668	0.830	0.770 ±0.131
	DSC	learned	0.954	0.974	0.955	0.955	0.964	0.961	0.966	0.967	0.963	0.951	0.961 ±0.007
		normal	0.955	0.972	0.949	0.953	0.965	0.955	0.961	0.970	0.969	0.960	0.961 ±0.008
GP	MMD	learned	1.303	0.995	1.145	1.276	1.024	0.884	1.070	1.126	1.123	0.993	1.090 ±0.138
		normal	1.4773	1.446	1.276	1.454	1.422	1.223	1.393	1.141	1.548	1.268	1.365 ±0.130
	DSC	learned	0.935	0.950	0.944	0.939	0.948	0.953	0.945	0.943	0.941	0.947	0.945 ±0.005
		normal	0.925	0.931	0.934	0.928	0.924	0.938	0.931	0.941	0.923	0.934	0.931 ±0.006

Table 10.1: Effect of learned shape priors on the accuracy of interactive segmentation of synthetic liver tumours in terms of mean minimum distance (MMD) and Dice similarity coefficient (DSC)

Model	Measure	Prior	User 1	User 2	User 3	User 4	User 5	User 6	User 7	User 8	User 9	User 10	Mean
Lan	MMD	learned	0.501	0.440	0.705	0.821	0.658	0.574	0.546	0.650	0.661	0.711	0.627 ±0.112
		normal	0.485	0.551	0.935	0.645	0.569	0.755	0.530	0.588	0.748	0.480	0.629 ±0.145
	DSC	learned	0.910	0.906	0.852	0.830	0.872	0.892	0.866	0.878	0.865	0.880	0.875 ±0.024
		normal	0.885	0.873	0.827	0.864	0.897	0.861	0.860	0.883	0.848	0.883	0.868 ±0.021
GP	MMD	learned	0.560	0.710	0.774	0.857	0.877	0.575	0.731	0.719	0.644	0.581	0.703 ±0.113
		normal	0.774	0.612	0.813	0.720	0.769	0.754	0.717	0.678	0.864	0.651	0.732 ±0.075
	DSC	learned	0.887	0.849	0.849	0.842	0.833	0.877	0.831	0.857	0.865	0.883	0.857 ±0.020
		normal	0.843	0.866	0.840	0.845	0.840	0.841	0.852	0.867	0.837	0.877	0.851 ±0.014

Table 10.2: Effect of learned shape priors on the accuracy of interactive segmentation of synthetic MS lesions in terms of mean minimum distance (MMD) and Dice similarity coefficient (DSC)

Model	Measure	Prior	User 1	User 2	User 3	User 4	User 5	User 6	User 7	User 8	User 9	User 10	Mean
Lan	MMD	learned	1.248	0.939	1.533	1.072	1.135	1.197	1.026	0.969	1.202	1.125	1.145 ±0.054
		normal	1.306	1.076	1.329	1.241	1.246	1.349	0.929	1.264	1.300	1.219	1.226 ±0.041
	DSC	learned	0.819	0.822	0.650	0.787	0.831	0.827	0.810	0.828	0.793	0.790	0.796 ±0.017
		normal	0.802	0.835	0.668	0.807	0.826	0.811	0.833	0.798	0.787	0.817	0.798 ±0.015
GP	MMD	learned	1.575	1.306	1.849	1.248	1.241	1.384	1.032	1.236	1.598	1.316	1.379 ±0.074
		normal	1.449	1.299	1.812	1.297	1.478	1.415	1.521	1.353	1.699	1.367	1.469 ±0.054
	DSC	learned	0.769	0.782	0.684	0.787	0.797	0.800	0.829	0.803	0.758	0.806	0.782 ±0.040
		normal	0.760	0.781	0.666	0.788	0.762	0.773	0.748	0.776	0.747	0.781	0.758 ±0.035

Table 10.3: Effect of learned shape priors on the accuracy of interactive segmentation of MRI MS lesions in terms of mean minimum distance (MMD) and Dice similarity coefficient (DSC)

Mod.	Meas.	Prior	Synthetic						MRI		
			liver tumour			MS lesion			MS lesion		
			i	ii	iii	i	ii	iii	i	ii	iii
Lan	MMD	learn	1.072±0.200	1.454±0.207	1.278±0.217	0.999±0.234	1.298±0.423	1.179±0.310	1.023±0.302	2.110±0.890	0.997±0.136
		norm	1.351±0.259	1.478±0.424	1.422±0.376	1.144±0.288	1.183±0.235	1.220±0.305	1.004±0.208	1.910±0.984	1.106±0.272
	DSC	learn	0.940±0.014	0.947±0.010	0.965±0.006	0.844±0.047	0.845±0.052	0.871±0.036	0.790±0.090	0.702±0.199	0.887±0.027
		norm	0.937±0.017	0.952±0.009	0.952±0.009	0.805±0.066	0.858±0.037	0.835±0.048	0.830±0.045	0.730±0.210	0.876±0.037
GP	MMD	learn	1.194±0.220	1.406±0.190	1.916±0.284	0.573±0.258	0.787±0.177	0.787±0.247	0.755±0.240	1.548±0.575	0.910±0.279
		norm	1.272±0.318	1.377±0.373	1.769±0.322	0.536±0.222	0.902±0.292	0.703±0.209	0.789±0.228	1.600±0.798	0.841±0.307
	DSC	learn	0.935±0.018	0.953±0.012	0.948±0.014	0.855±0.061	0.889±0.029	0.867±0.046	0.817±0.057	0.783±0.112	0.869±0.038
		norm	0.928±0.024	0.953±0.017	0.953±0.012	0.861±0.048	0.871±0.049	0.890±0.042	0.836±0.049	0.760±0.168	0.869±0.051

Table 10.4: Effect of learned shape priors on the inter-operator variability of interactive segmentation in terms of mean minimum distance (MMD) and Dice similarity coefficient (DSC)

Model	Measure	Prior	User 1	User 2	User 3	User 4	User 5	User 6	User 7	User 8	User 9	User 10	Mean
Lan	MMD	learned	3.101	1.026	2.547	2.135	1.114	1.444	1.015	1.534	2.686	1.459	1.806 ±0.756
		normal	0.947	0.975	1.092	1.435	0.871	1.725	1.293	1.412	1.137	2.178	1.307 ±0.404
	DSC	learned	0.954	0.967	0.955	0.966	0.966	0.956	0.966	0.973	0.969	0.955	0.963 ±0.007
		normal	0.957	0.970	0.964	0.960	0.975	0.948	0.958	0.970	0.963	0.949	0.961 ±0.009
GP	MMD	learned	1.959	1.270	1.832	1.450	1.768	1.801	2.114	1.339	2.023	1.476	1.703 ±0.299
		normal	1.849	2.096	1.400	1.852	1.687	1.841	1.911	1.125	2.563	1.925	1.825 ±0.384
	DSC	learned	0.942	0.964	0.960	0.958	0.948	0.947	0.937	0.972	0.951	0.966	0.955 ±0.011
		normal	0.948	0.946	0.968	0.957	0.962	0.952	0.940	0.976	0.919	0.956	0.952 ±0.016

Table 10.5: Effect of learned shape priors on the intra-operator variability of interactive segmentation of synthetic liver tumours in terms of mean minimum distance (MMD) and Dice similarity coefficient (DSC)

Model	Measure	Prior	User 1	User 2	User 3	User 4	User 5	User 6	User 7	User 8	User 9	User 10	Mean
Lan	MMD	learned	1.069	0.848	1.086	0.706	0.996	1.054	0.905	1.019	1.354	1.675	1.071 $\pm$0.272
		normal	1.003	0.992	1.246	0.916	0.996	1.070	0.939	1.024	0.891	1.154	1.023 $\pm$0.109
	DSC	learned	0.884	0.903	0.868	0.878	0.877	0.885	0.869	0.874	0.789	0.885	0.871 $\pm$0.031
		normal	0.892	0.860	0.856	0.875	0.880	0.871	0.896	0.877	0.849	0.856	0.871 $\pm$0.016
GP	MMD	learned	0.371	0.797	0.756	0.652	0.756	0.791	1.069	0.716	0.560	0.646	0.711 $\pm$0.180
		normal	0.527	0.610	0.210	1.290	0.646	0.862	0.702	0.541	0.777	0.610	0.678 $\pm$0.277
	DSC	learned	0.938	0.813	0.887	0.910	0.891	0.881	0.753	0.892	0.921	0.905	0.879 $\pm$0.055
		normal	0.911	0.880	0.977	0.790	0.899	0.873	0.895	0.917	0.891	0.906	0.894 $\pm$0.046

Table 10.6: Effect of learned shape priors on the intra-operator variability of interactive segmentation of synthetic MS lesions in terms of mean minimum distance (MMD) and Dice similarity coefficient (DSC)

Model	Measure	Prior	User 1	User 2	User 3	User 4	User 5	User 6	User 7	User 8	User 9	User 10	Mean
Lan	MMD	learned	0.836	0.870	1.399	0.931	0.810	0.894	0.701	0.628	0.723	1.207	0.900 $\pm$0.236
		normal	2.289	0.601	0.815	1.064	0.775	0.772	0.977	0.876	1.296	1.675	1.114 $\pm$0.515
	DSC	learned	0.917	0.876	0.811	0.906	0.921	0.900	0.901	0.925	0.916	0.914	0.899 $\pm$0.034
		normal	0.874	0.931	0.911	0.874	0.920	0.939	0.887	0.915	0.853	0.840	0.894 $\pm$0.034
GP	MMD	learned	0.803	1.099	1.005	0.596	0.887	0.666	0.656	0.360	0.691	0.125	0.689 $\pm$0.290
		normal	1.134	0.747	0.246	0.904	1.184	0.606	0.677	0.375	0.591	0.550	0.701 $\pm$0.302
	DSC	learned	0.898	0.858	0.857	0.924	0.851	0.898	0.903	0.934	0.905	0.983	0.901 $\pm$0.040
		normal	0.847	0.894	0.967	0.841	0.825	0.925	0.911	0.957	0.922	0.925	0.901 $\pm$0.049

Table 10.7: Effect of learned shape priors on the intra-operator variability of interactive segmentation of MRI MS lesions in terms of mean minimum distance (MMD) and Dice similarity coefficient (DSC)

model	Prior	User 1	User 2	User 3	User 4	User 5	User 6	User 7	User 8	User 9	User 10	Mean
Lan	learned	2.917	3.167	1.333	1.167	1.583	1.500	2.417	1.917	1.750	2.167	1.992 $\pm$0.670
	normal	3.000	3.250	1.167	1.917	2.167	1.750	4.167	2.917	1.917	2.250	2.450 $\pm$0.878
GP	learned	1.917	1.500	1.000	1.000	1.250	1.083	1.167	1.417	1.250	1.417	1.300 $\pm$0.278
	normal	1.583	2.000	1.000	1.250	1.667	1.333	2.000	1.667	1.250	1.333	1.508 $\pm$0.332

Table 10.8: Effect of learned shape priors on the number of times a contour model was initialised

Model	Measure	Prior	User 1	User 2	User 3	User 4	User 5	User 6	User 7	User 8	User 9	User 10	Mean
Lan	N_{int}	learned	2.500	21.250	5.000	14.250	12.250	10.000	11.250	5.750	6.000	17.000	10.520 $\pm$5.895
		normal	5.500	28.000	9.000	18.250	17.750	22.500	14.000	12.750	18.000	15.250	16.075 $\pm$6.450
	d_H	learned	3.555	2.962	4.971	4.485	4.827	4.220	3.559	3.100	4.321	8.083	4.413 $\pm$1.466
		normal	4.733	7.977	6.440	5.974	6.210	6.911	4.615	6.807	7.884	4.487	6.204 $\pm$1.273
GP	N_{int}	learned	2.250	5.7501	5.500	2.750	5.250	4.500	4.500	9.500	4.750	4.500	4.925 $\pm$1.958
		normal	7.500	9.000	9.000	11.250	7.000	9.500	13.750	16.250	6.500	9.000	9.875 $\pm$3.078
	d_H	learned	3.331	3.135	3.343	3.269	4.014	4.176	4.019	4.782	4.520	2.316	3.691 $\pm$0.742
		normal	4.450	4.190	3.272	2.722	4.075	4.180	5.589	4.767	5.579	3.800	4.262 $\pm$0.907

Table 10.9: Effect of learned shape priors on the level of post-editing necessary during interactive segmentation of synthetic liver tumours in terms of the number of interactions (N_{int}) and Hausdorff distance (d_H) between contours before and after post-editing

Model	Measure	Prior	User 1	User 2	User 3	User 4	User 5	User 6	User 7	User 8	User 9	User 10	Mean
Lan	N_{int}	learned	4.250	12.500	1.750	8.500	12.500	6.750	7.500	5.000	9.500	13.500	8.175 $\pm$3.900
		normal	6.750	12.250	4.0005	8.000	11.250	11.750	9.000	10.250	6.000	7.500	8.675 $\pm$2.708
	d_H	learned	2.081	2.894	2.140	2.179	3.493	3.393	2.193	3.871	3.427	1.927	2.760 $\pm$0.733
		normal	3.212	3.362	3.0504	2.163	3.159	3.454	2.727	2.460	3.374	3.087	3.005 $\pm$0.425
GP	N_{int}	learned	1.500	3.250	1.5006	0.750	1.2505	3.250	1.250	5.750	2.000	3.750	2.425 $\pm$1.550
		normal	3.500	4.500	2.500	1.000	2.750	6.250	3.250	8.750	1.311	2.750	3.656 $\pm$2.336
	d_H	learned	1.250	2.223	1.809	0.604	0.901	2.016	0.957	1.707	2.250	1.663	1.538 $\pm$0.579
		normal	1.559	2.168	1.500	0.854	1.350	2.340	0.913	1.894	2.057	1.516	1.615 $\pm$0.503

Table 10.10: Effect of learned shape priors on the level of post-editing necessary during interactive segmentation of synthetic MS lesions in terms of the number of interactions (N_{int}) and Hausdorff distance (d_H) between contours before and after post-editing

Model	Measure	Prior	User 1	User 2	User 3	User 4	User 5	User 6	User 7	User 8	User 9	User 10	Mean
Lan	N_{int}	learned	11.000	16.500	3.250	6.750	13.750	14.500	9.750	10.000	7.750	9.250	10.250 $\pm$3.926
		normal	5.250	16.500	3.750	9.750	16.750	18.500	19.000	9.500	8.750	8.000	11.575 $\pm$5.615
	d_H	learned	3.7186	3.460	2.472	1.766	2.460	3.632	3.027	2.668	2.753	1.766	2.772 $\pm$0.699
		normal	2.942	3.811	2.118	2.179	2.738	2.943	3.108	2.699	2.964	2.557	2.806 $\pm$0.484
GP	N_{int}	learned	3.000	5.250	2.000	3.750	1.500	2.500	2.500	4.000	1.750	1.000	2.725 $\pm$1.299
		normal	4.000	6.250	2.750	4.500	3.750	4.250	4.750	8.000	1.750	2.750	4.275 $\pm$1.808
	d_H	learned	2.759	2.4104	1.766	1.972	1.973	1.973	1.663	1.913	1.266	1.061	1.876 $\pm$0.493
		normal	1.809	2.475	1.663	2.121	1.914	1.913	2.266	1.516	2.505	1.604	1.979 $\pm$0.352

Table 10.11: Effect of learned shape priors on the level of post-editing necessary during interactive segmentation of MRI MS lesions in terms of the number of interactions (N_{int}) and Hausdorff distance (d_H) between contours before and after post-editing

Bibliography

[1] R.C. Gonzalez and R.E. Woods. *Digital Image Processing*. Pearson Prentice Hall, 2007.

[2] J. Fan, G. Zeng, M. Body, and M-S. Hacid. Seeded region growing: an extensive and comparative study. *Pattern Recognition Letters*, 26:1139–1156, 2005.

[3] B. Johnston, M.S. Atkins, B. Mackiewich, and M. Anderson. Segmentation of multiple sclerosis lesions in intensity corrected multispectral MRI. *Medical Imaging, IEEE Transactions on*, 15:154–169, 1996.

[4] R. M. Haralick and L. G. Shapiro. Image segmentation techniques. *Proceedings of SPIE - The International Society for Optical Engineering*, 548:2–9, 1985.

[5] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. In *Proceedings, IEEE International Conference on Computer Vision*, pages 321–330, 1987.

[6] R. Malladi, J. A. Sethian, and B. C. Vemuri. Shape modeling with front propogation: A level set approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:158–175, 1995.

[7] C. Xu and J.L. Prince. Gradient vector flow: A new external force for snakes. In *Proceedings, IEEE Conference on Computer Vision and Pattern Recognition*, pages 66–71, 1997.

[8] J. Ivins and J. Porrill. Active region models for segmenting medical images. In *IEEE first international conference on Image Processing*, pages 227–231, 1994.

[9] T. F. Cootes and C. Taylor. Active shape models – smart snakes. In *Proceedings, British Machine Vision Conference, BMVA Press*, pages 266–276, 1992.

[10] J. Wang, A. Ekin, and G. de Haan. Shape analysis of brain ventricles for improved classification of alzheimer's patients. In *Proceedings, IEEE International Conference on Image Processing*, pages 2252–2255, 2008.

[11] R. P. Grzeszczuk and D. N. Levin. Brownian strings: Segmenting images with stochastically deformable contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:1100–1114, 1997.

[12] D.J. Williams and M. Shah. A fast algorithm for active contours and curvature estimation. *Computer Vision, Graphics and Image Processing – Image Understanding*, 55:14–26, 1992.

[13] J. Stawiaski, E. Decencire, and F. Bidault. Interactive liver tumor segmentation using graph cuts and watershed. In *Proceedings, Medical Image Computing and Computer-Assisted Intervention: Workshop on 3D Segmentation in the Clinic*, 2008.

[14] Y. Ge. Multiple sclerosis: The role of MR imaging. *American Journal Of Neuroradiology*, 27:1165–1176, 2006.

[15] H. Lassmann, W. Bruck, and C. Lucchinetti. Heterogeneity of multiple sclerosis pathogenesis: Implications for diagnosis and therapy. *Trends in Molecular Medicine*, 7:115–121, 2001.

[16] C. Pachai, Y.M. Zhu, J. Grimaud, M. Hermier, A. Dromigny-Badin, A. Boudraa, G. Gimenez, C. Confavreux, and J.C. Froment. A pyramidal approach for automatic segmentation of multiple sclerosis lesions in brain MRI. *Computerized Medical Imaging and Graphics*, 22:399–408, 1998.

[17] C. Lucchinetti, W. Bruck, J. Parisi, B. Scheithauer, H. Rodriguez, and H. Lassmann. Heterogeneity of multiple sclerosis lesions: Implications for the pathogenesis of demyelination. *Annals of Neurology*, 47:707–717, 2000.

[18] J.H. Simon, A. Sherzinger, U. Raff, and X.Li. Computerized method of lesion volume quantitation in multiple sclerosis: Error of serial studies. *American Journal Of Neuroradiology*, 18:580–582, 1996.

[19] Dr. Jo Swanton. *in conversation.* Institute of Neurology, London, 2006.

[20] Dr. M. Horsman. *in correspondence.* Leicester University, 2007.

[21] Dr. JF. Barkhof. *in correspondence.* VUMC, Amsterdam, 2007.

[22] D.L. Plummer. Dispimage: Un mezzo di analisi e presentazione per iconografia medica. *Riv. Neuroradiol*, 5:489–495, 1992.

[23] Dr. Jo Swanton. *in conversation.* Institute of Neurology, London, 2007.

[24] Dr. Tom Hayton. *in conversation.* Institute of Neurology, London, 2007.

[25] Dr. Julian Furby. *in conversation.* Institute of Neurology, London, 2007.

[26] Dr. Leonora Fisniku. *in conversation.* Institute of Neurology, London, 2007.

[27] D. M. Cash, M. I. Miga, S. C. Glasgow, B. M. Dawant, L. W. Clements, Z. Cao, R. L. Galloway, and W. C. Chapman. Concepts and preliminary data toward the realization of image-guided liver surgery. *Journal of Gastrointestinal Surgery*, 11:844–859, 2007.

[28] K. Zavadsky and Y. lee. Liver metastases from colorectal carcinoma: incidence, resectability, and survival results. *The American Surgeon*, 60:929–933, 1994.

[29] T. Heimann and B. van Ginneken. 3D liver tumor segmentation challenge 2008, part of *3D segmentation in the clinic: A Grand Challenge II*, at 11^{th} international conference on Medical Image Computing and Computer Assisted Intervention, New York, 2008.

[30] P. J. Davis, E. A. Kosmacek, Y. Sun, F. Ianzini, and M. A. Mackey. The large scale digital cell analysis system: An open system for non-perturbing live cell imaging. *Journal of Microscopy*, 228:269–308, 2007.

[31] I. Ersoy, F. Bunyak, M. A. Mackey, and K. Palaniappan. Cell segmentation using hessian-based detection and contour evolution with directional derivatives. In *Proceedings, IEEE International Conference on Image Processing*, pages 1804–1807, 2008.

[32] W. Denck and H. Horstmann. Serial block-face scanning electron microscopy to reconstruct three-dimensional tissue nanostructure. *PLoS Biology*, 2:e329, 2004.

[33] N. Vu and B. S. Manjunath. Graph cut segmentation of neuronal structures from transmission electron micrographs. In *Proceedings, IEEE International Conference on Image Processing*, pages 725–728, 2008.

[34] J. H. Macke., N. Maack, R. Gupta, W. Denck, B. Schölkopf, and A. Borst. Contour-propagation algorithms for semi-automated reconstruction of neuronal processes. *Journal of Neuroscience Methods*, 167:349–357, 2008.

[35] N. C. Fox, P. A. Freeborough, and M. N. Rossor. Visualisation and quantification of rates of atrophy in alzheimer's disease. *LANCET*, 9020:94–97, 1996.

[36] P.A. Freeborough, N.C. Fox, and R.I. Kitney. Interactive algorithms for the segmentation and quantitation of 3D MRI brain scans. *Computer Methods and Programming in Biomedicine*, 53:15–25, 1997.

[37] Dr. Jo Foster. *in conversation*. Institute of Neurology, London, 2006.

[38] L. H. Staib and J. S. Duncan. Model-based deformable surface finding for medical images. *IEEE Transactions on Medical Imaging*, 15:720–731, 1996.

[39] J. K. Udupa, V. R. LeBlanc, Y. Zhuge, C. Imielinska, H. Schmidt, L. M. Currie, B. E. Hirsch, and J. Woodburn. A framework for evaluating image segmentation algorithms. *Computerized Medical Imaging and Graphics*, 30:75–87, 2006.

[40] J. Liang, T. McInerney, and D. Terzopoulos. United snakes. *Medical Image Analysis*, 10:215–233, 2006.

[41] A.X. Falcão, J.K. Udupa, and F.K. Miyazawa. An ultra–fast user–steered image segmentation paradigm: Live wire on the fly. *IEEE Transactions on Medical Imaging*, 19:55–62, 2000.

[42] A.X. Falcão, J.K. Udupa, S. Samarasekera, and S. Sharma. User–steered image segmentation paradigms: Live wire and live lane. *Graphical Models and Image Processing*, 60:233–260, 1998.

[43] P. Pérez, A. Blake, and M. Gangnet. Jetstream: Probabilistic contour extraction with particles. In *Proceedings, IEEE International Conference on Computer Vision*, pages 524–531, 2001.

[44] Media Laboratory, M.I.T. Vision Texture 1.0. http://www−white.media.mit.edu/vismod/imagery/VisionTexture/vistex.html.

[45] M. Jacob, T. Blu, and M. Unser. Efficient energies and algorithms for parametric snakes. *IEEE Transactions on Image Processing*, 13:1231–1244, 2004.

[46] M. Fradkin, C. Ciofolo, B. Mory, G. Hautvast, and M. Breeuwer. Comprehensive segmentation of cine cardiac MR images. In *Proceedings, Medical Image Computing and Computer-Assisted Intervention*, pages 178–185, 2008.

[47] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contours. *International Journal of Computer Vision*, 22:61–79, 1997.

[48] J. Gomes and O. Faugeras. Level sets and distance functions. In *Proceedings, European Conference on Computer Vision*, pages 588–602, 2000.

[49] J.S Suri, K. Liu, S. Singh, S.N. Laxminarayan, X. Zeng, and L. Reden. Shape recovery algorithms using level sets in 2D / 3D medical imagery: a state-of-the-art review. *IEEE Transactions on Information Technology in Biomedicine*, 6:8–28, 2002.

[50] M. Leventon, E. Grimson, and O. Faugeras. Statistical shape influence in geodesic active contours. In *Proceedings, IEEE Conference on Computer Vision and Pattern Recognition*, pages 316–323, 2000.

[51] Y. Chen, H. D. Tagare, S. Thiruvenkadam, F. Huang, D. Wilson, K. S. Gopinas, R. W. Briggs, and E. A. Geiser. Using prior shapes in geometric active contours in a variational framework. *International Journal of Computer Vision*, 50:315–328, 2002.

[52] X. Bresson, P. Vandergheynst, and J-P. Thiran. A variational model for object segmentation using boundary information and shape prior driven by the mumford-shah functional. *International Journal of Computer Vision*, 68:145–162, 2006.

[53] N. Houhou, A. Lemkaddem, V. Duay, Abdelkarim Allal, and J.-P. Thiran. Shape prior based on statistical map for active contour segmentation. In *Proceedings, IEEE International Conference on Image Processing*, pages 2284–2287, 2008.

[54] H. Shen, A. Litvin, and C. Alvino. Localized priors for the precise segmentation of individual vertebras from CT volume data. In *Proceedings, Medical Image Computing and Computer-Assisted Intervention*, pages 367–375, 2008.

[55] O. Juan, R. Keriven, and G. Postel. Stochastic motion and the level set method in computer vision: Stochastic active contours. *International Journal of Computer Vision*, 69:7–25, 2006.

[56] A. J. Fan, J. Fisher, W. Wells, J. Levitt, and A. Willsky. MCMC curve sampling for image segmentation. In *Proceedings, Medical Image Computing and Computer-Assisted Intervention*, 2007.

[57] A.A. Amini, S. Tehrani, and T.E. Weymouth. Using dynamic programming for minimizing the energy of active contours in the presence of hard constraints. In *Proceedings, IEEE International Conference on Computer Vision*, pages 95–99, 1988.

[58] X.M. Pardo, M.J. Carreira, A. Mosquera, and D. Cabello. A snake for CT image segmentation integrating region and edge information. *Image and Vision Computing*, 19:461–475, 2001.

[59] KF Lai and RT Chin. On regularisation, formulation and initialisation of the active contour models (snakes). In *Proceedings, Asian Conference on Computer Vision*, pages 542–545, 1993.

[60] D.P. Perrin and C.E. Smith. Rethinking classical internal forces for active contour models. In *Proceedings, IEEE Conference on Computer Vision and Pattern Recognition*, 2001.

[61] X. Ran and F. Xi. Segmental active contour model integrating region information for medical image segmentation. In *Proceedings, IEEE Conference on Medical Imaging and Augmented Reality*, pages 129–136, 2004.

[62] I. Dindoyal, T. Lambrou, J. Deng, C.F. Ruff, A.D. Linney, and A. Todd-Pokropek. An active contour model to segment foetal cardiac ultrasound data. In *Proceedings, Medical Image Understanding and Analysis*, pages 77–80, 2003.

[63] J. Ivins and J. Porrill. Statistical snakes: Active region models. In *Proceedings, British Machine Vision Conference, BMVA Press*, pages 377–386, 1994.

[64] R. Ronfard. Region based strategies for active contour models. *International Journal of Computer Vision*, 13:229–251, 1994.

[65] J. Ivins and J. Porrill. Constrained active region models for fast tracking in colour image sequences. *Computer Vision and Image Understanding*, 72:54–71, 1998.

[66] A. Marin-Hernandez, M. Devy, and G. Avina-Cervañtes. Colour active contours for tracking roads in natural environments. In *Proceedings, Iberoamerican Congress on Pattern Recognition*, pages 124–131, 2004.

[67] D.C. Alexander and B.F. Buxton. Statistical modeling of colour data. *International Journal of Computer Vision*, 44:87–109, 2001.

[68] J. Ivins and J. Porrill. Active region models for segmenting textures and colours. *Image and Vision Computing*, 13:431–438, 1995.

[69] J. Marti, J. Freixenet, X. Muñoz, and A. Oliver. Active region segmentation of mammographic mass based on texture and shape features. In *Proceedings, Iberian Conference on Pattern Recognition and Image Analysis*, pages 478–485, 2003.

[70] O. Pujol and P. Radeva. Texture segmentation by statistical deformable models. *International Journal of Image and Graphics*, 4:433–452, 2004.

[71] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[72] N. S. Friedland and D. Adam. Ventricular cavity boundary detection from sequential ultrasound images using simulated annealing. *IEEE Transactions on Medical Imaging*, 8, 1989.

[73] N. S. Friedland and A. Rosenfeld. Compact object recognition using energy-function-based optimisation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14, 1992.

[74] J. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society*, 36:192–225, 1974.

[75] M. A. T. Figueiredo and J. M. N. Leitão. Bayesian estimation of ventricular contours in angiographic images. *IEEE Transactions on Medical Imaging*, 11:416–429, 1992.

[76] J. Dias and J. Leitao. Wall position and thickness estimation from sequences of echocardiographic images. *IEEE Transactions on Medical Imaging*, 15, 1996.

[77] M. Martin and C. Alberola. A bayesian approach to in vivo kidney ultrasound contour detection using Markov random fields. In *Proceedings, Medical Image Computing and Computer-Assisted Intervention*, pages 397–4043, 2002.

[78] M. Martín-Fernández and C. Alberola-López. An approach for contour detection of human kidneys from ultrasound images using Markov random fields and active contours. *Medical Image Analysis*, 9, 2005.

[79] M. Loog and B. van Ginneken. Supervised segmentation by iterated contextual pixel classification. In *ICPR*, pages 925–928, 2002.

[80] M. Rivera and P. Mayorga. Quadratic Markovian probability fields for image binary segmentation. In *Proceedings, IEEE International Conference on Computer Vision*, pages 1–8, 2007.

[81] J. Zhao and X. Geng. Interactive image segmentation via multi-cue dynamic integration. In *Proceedings, IEEE International Conference on Image Processing*, pages 3044–3047, 2008.

[82] A. Rosenfeld and A. C. Kak. *Digital Picture Processing*. Academic Press, Orlando, 1982.

[83] F. O'Sullivan, S. Roy, J. O'Sullivan, C. Vernon, and J. Eary. Incorporation of tumor shape into an assessment of spatial heterogeneity for human sarcomas imaged with FDG-PET. *Biostatistics*, 6:293–301, 2005.

[84] D. Geman and B. Jedynak. Shape recognition and twenty questions. Technical Report 2155, INRIA-Rocquencourt, 1993.

[85] D. Geman and B. Jedynak. An active testing model for tracking roads in satellite images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18:1–14, 1996.

[86] M. Isard and A. Blake. CONDENSATION - Conditional probability density propagation for visual tracking. *International Journal of Computer Vision*, 29:5–28, 1998.

[87] Y.E. Famao, S.U. Lin, L. Shukai, and L. Shengyang. Extraction of complex object contour by particle filtering. In *IEEE Int. Geoscience and Remote Sensing Symposium*, pages 3711–3713, 2005.

[88] K. Allen, C. Yau, and A. Noble. A recursive, stochastic vessel segmentation framework that robustly handles bifurcations. In *Proceedings, Medical Image Understanding and Analysis*, pages 19–23, 2008.

[89] E.M. Mortensen, B.S. Morse, W.A. Barrett, and J.K. Udupa. Adaptive boundary detection using live wire two dimensional dynamic programming. In *Proceedings, Compuers in Cardiology*, pages 635–638, 1992.

[90] E.M. Mortensen and W.A. Barrett. Intelligent scissors for image composition. In *Proceedings, ACM SIGGRAPH*, pages 191–198, 1995.

[91] J. S. Park, M. S. Chung, S. B. Hwang, Y. S. Lee, and D-H. Har. Technical report on semiautomatic segmentation using the Adobe Photoshop. *Journal of Digital Imaging*, 18:333–343, 2005.

[92] J. Grimaud, M. Lai, and J. Thorpe. Quantification of MRI lesion load in multiple sclerosis: A comparison of three different computer-assisted techniques. *Magnetic Resonance Imaging*, 14:495–505, 1996.

[93] J. Luan, J. Stander, and D. Wright. On shape detection in noisy images with particular reference to ultrasonography. *Statistics and Computing*, 8:14 – 22, 1998.

[94] R. Adams and L. Bischof. Seeded region growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16:641–647, 1994.

[95] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum. Lazy snapping. In *Proceedings, ACM SIGGRAPH*, pages 303–308, 2004.

[96] C. Rother, V. Kolmogorov, and A Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. In *Proceedings, ACM SIGGRAPH*, pages 309–314, 2004.

[97] J. Wang, P. Bhat, R.A Colburn, M.Agrawala, and M.F. Cohen. Interactive video cutout. In *Proceedings, ACM SIGGRAPH*, pages 585–594, 2005.

[98] X. Yuan, N. Zhang, M.X. Nguyen, and B. Chen. Volume cutout. *The Visual Computer*, 21:745–754, 2005.

[99] Y. Boykov and M. P. Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in n-d images. In *Proceedings, IEEE International Conference on Computer Vision*, 2001.

[100] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:1222–1239, 2001.

[101] R. A. Robb and D. P. Hanson. ANALYZE: a software system for biomedical image analysis. In *Proceedings, Visualisation in Biomedical Computing*, pages 507–518, 1990.

[102] J. M. P. Lötjönen, V. M. Järvinen, B. Cheong, E. Wu, S. Kivistö, J. R. Koikkalainen, J. J. O. Mattila, H. M. Kervinen, R. Muthupillai, F. H. Sheehan, and K. Lauerma. Evaluation of cardiac biventricular segmentation from multiaxis MRI data: A multicenter study. *Journal of Magnetic Resonance Imaging*, 28:626–636, 2008.

[103] T. Heimann, M. Thorn, T. Kunert, and H.P. Meinzer. New methods for leak detection and contour correction in seeded region growing segmentation. *International Archives of Photogrammetry and Remote Sensing*, 35, 2004.

[104] L. R. Dice. Measures of the amount of ecologic association between species. *Ecology*, 26:97//302, 1945.

[105] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis*. Wiley, 1973.

[106] D.P. Huttenlocher, G.A. Klanderman, and W.J. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15:850–863, 1993.

[107] M. Dubuisson and A. K. Jain. A modified Hausdorff distance for object matching. In *Proceedings, International Conference on Pattern Recognition*, pages 566–568, 1994.

[108] B. Takács. Comparing face images using the modified hausdorff distance. *Pattern Recognition*, 31:1873–1881, 1998.

[109] Y. Rogers, H Sharp, and J. Preece. *Interaction Design - Beyond human-computer interaction*. Wiley, second edition, 2002.

[110] J. P. Chin, V. A. Diehl, and K. L. Norman. Development of an instrument measuring user satisfaction of the human-computer interface. In *Proceedings, SIGCHI conference on Human Factors in Computing Systems*, pages 213–218, 1988.

[111] A. Bulpitt and N. Efford. An efficient 3D deformable model with a self-optimising mesh. *Image and Vision Computing*, 14:573–580, 1996.

[112] J. Bredno, T.M. Lehmann, and K. Spitzer. A general discrete contour model in two, three, and four dimensions for topology-adaptive multichannel segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25, 2003.

[113] A. Evans, T. Lambrou, A. D. Linney, and A. Todd-Pokropek. Automatic 3D segmentation of the liver from computed tomography images, a discrete deformable model approach. In *Proceedings, International Conference on Control, Automation, Robotics and Vision*, pages 1–6, 2006.

[114] L. D. Cohen and I. Cohen. Deformable models for 3D medical images using finite elements and balloons. In *Proceedings, IEEE Conference on Computer Vision and Pattern Recognition*, pages 592–598, 1992.

[115] A. Schenk, G. Prause, and H.-O. Pietgen. Efficient semiautomatic segmentation of 3D objects in medical images. In *Proceedings, Medical Image Computing and Computer-Assisted Intervention*, pages 186–195, 2000.

[116] L.H. Staib and J.S. Duncan. Boundary finding with parametrically deformable models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:161–175, 1992.

[117] D. Cremers, S. J. Osher, and S. Soatto. Kernel density estimation and intrinsic alignment for shape priors in level set segmentation. *International Journal of Computer Vision*, 69:335–351, 2006.

[118] T. F. Cootes and C. J. Taylor. Combining point distribution models with shape. *Image and Vision Computing*, 13:403–409, 1995.

[119] M. deBruijne and M. Nielsen. Shape particle filtering for image segmentation. In *Proceedings, Medical Image Computing and Computer-Assisted Intervention*, pages 168–175, 2004.

[120] S. M. Pizer, T. Fletcher, Y. Fridman, D. S. Fritsch, A. G. Gash, J. M. Glotzer, S. Joshi, A. Thall, G. Tracton, P. Yushkevich, and E. L. Chaney. Deformable M-reps for 3D medical image segmentation. *International Journal of Computer Vision*, 55:85–106, 2003.

[121] Z. Tu, K. L. Narr, P. Dollár, I. Dinov, P. M. Thompson, and A. W. Toga. Brain anatomical structure segmentation by hybrid discriminative/generative models. *IEEE Transactions on Medical Imaging*, 27, 2008.

[122] E. B. Dam, P. T. Fletcher, and Stephen M. Pizer. Automatic shape model building based on principal geodesic analysis bootstrapping. *Medical Image Analysis*, 12:136–151, 2008.

[123] A. Hill and C. J. Taylor. Automatic landmark generation for point distribution models. In *Proceedings, British Machine Vision Conference*, pages 429–438, 1994.

[124] A. Hill, C. J. Taylor, and A. D. Brett. A framework for automatic landmark identification using a new method of nonrigid correspondence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:241–251, 2000.

[125] D. Rueckert, A. Frangi, and J. Schnabel. Automatic construction of 3D statistical deformation models of the brain using nonrigid registration. *IEEE Transactions on Medical Imaging*, 22:1014–1025, 2003.

[126] C. Twining, T. F. Cootes, S. Marsland, V. Petrovic, R Schestowitz, and C. Taylor. A unified information-theoretic approach to group-wise nonrigid registration and model building. In *Proceedings, Information Processing in Medical Imaging*, volume 3565, 2005.

[127] R. H. Davies, C. J. Twining, T. F. Cootes, J. C. Waterton, and C. J. Taylor. A minimum description length approach to statistical shape modeling. *IEEE Transactions on Medical Imaging*, 21:525–537, 2002.

[128] R. H. Davies, C. J. Twining, T .F. Cootes, J. C. Waterton, and C. J. Taylor. 3D statistical shape models using direct optimisation of description length. In *Proceedings, European Conference on Computer Vision*, volume 3, pages 3–20, 2002.

[129] J. R. Rissanen. A universal prior for integers and estimation by minimal description length. *Annals of Statistics*, 11:416–431, 1983.

[130] M. Berks, S. Caulkin, R. Rahim, C. Boggis, and S. Astley. Statistical appearance models of mammographic masses. *Digital Mammography (LNCS)*, 5116:401–408, 2008.

[131] S. Caulkin, S. Astley, A. Mills, and C. Boggis. Generating realistic spiculated lesions in digital mammograms. In *Proceedings, International Workshop on Digital Mammography*, pages 713–720, 2000.

[132] E. Fatemizadeh, C. Lucas, and H. Soltanian-Zadeh. Automatic landmark extraction from image data using modified growing neural gas network. *IEEE Transactions on Information Technology in Biomedicine*, 7:77–85, 2003.

[133] T. Kohonen. The self-organising map. *Proceedings of the IEEE*, 78:1464–1480, 1990.

[134] C. H. Teh and R. T. Chin. On the detection of dominant points on digital curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11, 1989.

[135] D. Cremers, T. Kohlberger, and C. Schnörr. Nonlinear shape statistics in Mumford-Shah based segmentation. In *Proceedings, European Conference on Computer Vision*, pages 93–108, 2002.

[136] B. Chalmond and S. C. Girard. Nonlinear modelling of scattered multivariate data and its application to shape change. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21:422–432, 2002.

[137] T. F. Cootes and C. J. Taylor. A mixture model for representing shape variation. *Image and Vision Computing*, 17:567–574, 1999.

[138] D. Cremers, T. Kohlberger, and C. Schnörr. Shape statistics in kernel space for variational image segmentation. *Pattern Recognition*, 36:1929–1943, 2003.

[139] L. Mei, M. Figl, D. Rueckert, A. Darzi, and P. Edwards. Sample sufficiency and number of modes to retain in statistical shape modelling. In *Proceedings, Medical Image Computing and Computer-Assisted Intervention*, pages 425–433, 2008.

[140] Y. Shi and D. Shen. Hierarchical shape statistical model for segmentation of lung fields in chest radiographs. In *Proceedings, Medical Image Computing and Computer-Assisted Intervention*, pages 417–424, 2008.

[141] M. de Bruijne, M. T. Lund, L. B. Tanko, P. P. Pettersen, and M. Nielsen. Quantitative vertebral morphometry using neighbor-conditional shape models. In *Proceedings, Medical Image Computing and Computer-Assisted Intervention*, pages 1–8, 2006.

[142] T. Klinder, R. Wolz, C. Lorenz, A. Franz, and J. Ostermann. Spine segmentation using articulated shape models. In *Proceedings, Medical Image Computing and Computer-Assisted Intervention*, pages 227–234, 2008.

[143] C. Xu, A. Yezzi, and J. L. Prince. On the relationship between parametric and geometric active contours. In *Proceedings, Asilomar Conference on Signal, Systems and Computers*, pages 483–489, 2000.

[144] G. Hamarneh and T. Gustavsson. Combining snakes and active shape models for segmenting the human left ventricle in echocardiographic images. *Computers in Cardiology*, 27:115–118, 2000.

[145] H. Devlin, P. D. Allen, J. Graham, R. Jacobs, K. Karayianni, C. Lindh, P. F. van der Stelt, E. Harrison, J. E. Adams, S. Pavitt, and K. Horner. Automated osteoporosis risk assessment by dentists: A new pathway to diagnosis. *Bone*, pages 835 – 842, 2007.

[146] R. Pinho, J. Sijbers, and T. Huysmans. Segmentation of the human trachea using deformable statistical models of tubular shapes. In *Advanced Concepts for Intelligent Vision Systems*, pages 531–542. Springer Berlin, 2007.

[147] J. Hug, C. Brechb uhler, and G. Székely. Model-based initialisation for segmentation. In *Proceedings, European Conference on Computer Vision*, pages 290–306, 2000.

[148] B. van Ginneken, M. de Bruijne, M. Loog, and M. A. Viergever. Interactive shape models. In *Proceedings, SPIE Medical Imaging*, pages 1206–1216, 2003.

[149] B. van Ginneken, M. B. Stegmann, and M. Loog. Segmentation of anatomical structures in chest radioghraphs using supervised methods: a comparative study on a public database. *Medical Image Analysis*, 10:19–40, 2006.

[150] S. Pizer, D. Fritsch, P. Yushkevich, V. Johnson, and E. Cheney. Segmentation, registration and measurement of shape variation via image object shape. *IEEE Transactions on Medical Imaging*, 18:851–865, 1999.

[151] P. Yushkevich, H. Zhang, and J. C. Gee. Statistical modeling of shape and appearance using the continuous medial representation. In *Proceedings, Medical Image Computing and Computer-Assisted Intervention*, pages 725–732, 2005.

[152] J. H. Levy, K. Gorczowski, X. Liu, and S. M. Pizer. Caudate segmentation using deformable M-reps. In *Proceedings, Medical Image Computing and Computer-Assisted Intervention*, 2007.

[153] K. Siddiqi and S. M. Pizer. *Medial Representations: Mathematics Algorithms and Applications*. Computational Imaging 37, Springer, 2007.

[154] ”M. A. Styner, K. T. Rajamani, L-P. Nolte, G. Zsemlye, and G. Székely”. Evaluation of 3D correspondence methods for model building. In *Proceedings, Information Processing in Medical Imaging*, pages 63–75, 2003.

[155] M Styner, G Gerig, S Joshi, and S Pizer. Automatic and robust computation of 3D medial models incorporating object variability. *International Journal of Computer Vision*, 55:107–122, 2003.

[156] S. M. Pizer. The medical image display and analysis group at the university of north carolina: Reminiscences and philosophy. *IEEE Transactions on Medical Imaging*, 22:2–10, 2003.

[157] M.Kaus, K. Brock, V. Pekar, L. Dawson, A. Nichol, and D. Jaffray. Assessment of a model based deformable image registration approach for radiation treatment planning. *International Journal of Radiation Oncology Biology and Physics*, 68:572–580, 2007.

[158] R. L. Kashyap and R. Chellappa. Stochastic models for closed boundary analysis: Representation and reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:627–637, 1981.

[159] K. Eom and J. Park. Recognition of shapes by statistical modelling of centroidal profile. In *Proceedings, International Conference on Pattern Recognition*, pages 860–864, 1990.

[160] H. Kauppinen, T. Seppanen, and M. Pietikainen. An experimental comparison of autoregressive and Fourier-based descriptors in 2D shape classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17, 1995.

[161] M. Das, F. Butterworth, and R. Das. Statistical signal modeling techniques for automated recognition of water-borne microbial shapes. In *Proceedings, IEEE Symposium on Circuits and Systems*, pages 613–616, 1997.

[162] A. H. Mir, M. Hanmandlu, and S. N. Tandon. Description of shapes in CT images. *IEEE Engineering in Medicine and Biology*, 18:79–84, 1999.

[163] S. R. Dubois and F. H. Glanz. An autoregressive model approach to two-dimensional shape classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8:55–66, 1986.

[164] M. Das, M. J. Paulik, and N. K. Loh. A bivariate autoregressive modeling technique for analysis and classification of planar shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:97–103, 1990.

[165] B. Kartikeyan and A. Sarkar. Shape description by time series. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11:977–984, 1989.

[166] Y. He and A. Kundu. 2-D shape classification using hidden Markov model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:1172–1184, 1991.

[167] L. R. Rabiner, J. G. Wilpon, and B. J. Juang. A segmental k-means training procedure for connected word recognition. *AT & T Technical Journal*, 65:21–31, 1986.

[168] D. G. Forney. The Viterbi algorithm. In *Proceedings, IEEE*, pages 263–278, 1973.

[169] F. P. Kuhl and C. R. Giardina. Elliptic Fourier features of a closed contour. *Computer Graphics and Image Processing*, 18:236–258, 1982.

[170] G. H. Granlund. Fourier preprocessing for hand print character recognition. *IEEE Transactions on Computers*, C-21:195–201, 1972.

[171] O. R. Mitchell and T. A. Grogan. Global and partial shape discrimination for computer vision. *Optical Engineering*, 23:484–491, 1984.

[172] G. Szekély. Segmentation of 2-D and 3-D objects from MRI volume data using constrained elastic deformations of flexible Fourier contour and surface models. *Medical Image Analysis*, 1, 1996.

[173] M-K. Hu. Visual pattern recognition by moment invariants. *IEEE Transactions on Information Theory*, 8:197–187, 1962.

[174] M. Bober. MPEG-7 visual shape descriptors. *IEEE Transactions on Circuits and Systems for Video Technology*, 11:716–719, 2001.

[175] L. Gorelick, M. Galun, E. Sharon, R. Basri, and A. Brandt. Shape representation and classification using the Poisson equation. In *Proceedings, IEEE Conference on Computer Vision and Pattern Recognition*, pages 61–67, 2004.

[176] Maria Petrou. *BMVA symposium on Machine Learning.* London, 29 October 2008.

[177] A.K. Jain. *Image Analysis and Computer Vision*, chapter 9, pages 342–425. Fundamentals of Digital Image Processing. Prentice Hall, 1989.

[178] T. Hastie, R. Tibshirhani, and A. Buja. Flexible discriminant analysis by optimal scoring. *Journal of the American Statistical Association*, 89:1255–1270, 1994.

[179] G. Baudat and F. Anouar. Generalized discriminant analysis using a kernel approach. *Neural computation*, 12:2385–2404, 2000.

[180] S. Billings and K. Lee. Nonlinear fisher discriminant analysis using a minimum squared error cost function and the orthogonal least squares algorithm. *Neural networks*, 15:263–270, 2002.

[181] V. Roth and V. Steinhage. Nonlinear discriminant analysis using kernel functions. *Advances in Neural Information Processing Systems*, 12:568–574, 2000.

[182] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and Systems Sciences*, 55:119–139, 1997.

[183] I. Guyon, B.E. Boser, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings, Workshop of Computer Learning Theory*, pages 144–152, 1992.

[184] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.

[185] V. N. Vapnik. *Estimation of Dependences based on Empirical Data*. Springer-Verlag, 1982.

[186] E. Osuna, R. Freund, and F. Girosi. Improved training algorithm for support vector machines. In *IEEE Workshop on Neural Networks for Signal Processing*, 1997.

[187] J. C. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Scholkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 185–208. Cambridge MA:MIT Press, 1998.

[188] R.-E. Fan, P.-H. Chen, and C.-J. Lin. Working set selection using second order information for training SVM. *Journal of Machine Learning Research*, 6:1889–1918, 2005.

[189] J. Yao, R.M. Summers, and A. Hara. Optimizing the support vector machines committee configuration in a colonic polyp cad system. In *Proceedings, SPIE Medical Imaging*, pages 384–392, 2005.

[190] T.G. Dietterich and G. Bakari. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

[191] M. Kugler, H. Matsuo, and A. Iwata. A new approach for applying support vector machines in multiclass problems using class groupings and truth tables. In *Proceedings, Pacific Rim International Conference on Artificial Intelligence*, pages 1013–1014, 2004.

[192] S. Godbole, S. Sarawagi, and S. Chakrabarti. Scaling multi-class support vector machines using inter-class confusion. In *Proceedings, ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 513–518, 2002.

[193] J. Weston and C. Watkins. Support vector machines for multi–class pattern recognition. In *Proceedings, European Symposium On Artificial Neural Networks*, pages 219–224, 1999.

[194] C-W. Hsu and C-J. Lin. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13:415–425, 2002.

[195] G. M. Foody and A. Mathur. A relative evaluation of multiclass image classification using support vector machines. *IEEE Transactions on Geoscience and Remote Sensing*, 42:1335–1343, 2004.

[196] J. C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In A. Smola, P. Barlett, B. Schölkopf, and D Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 61–74. Cambridge MA:MIT Press, 1999.

[197] C-W. Hsu, C-C. Chang, and C-J. Lin. A practical guide to support vector classification. Technical report, Department of Computer Science and Information Engineering, National Taiwan University, Taipei, 2003.

[198] K.I. Kim, K. Jung, S.H. Park, and H.J. Kim. Supervised texture segmentation using support vector machines. *Electronics Letters*, 35:1935–1937, 1999.

[199] K.I. Kim, K. Jung, S.H. Park, and H.J. Kim. Support vector machines for texture classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:1542–1550, 2002.

[200] J. Mao and A. K. Jain. Texture classification and segmentation using multiresolution simultaneous autoregressive models. *Pattern Recognition*, pages 173–188, 1992.

[201] K. Jung and J.H. Han. Texture-based text location for video indexing. In *Proceedings, Intelligent Data Engineering and Automated Learning*, pages 449–454, 2000.

[202] R. Campanini, D. Dongiovanni, E. Iampieri, N. Lanconelli, M. Masotti, G. Palermo, A. Riccardi, and M. Roffelli. A novel featureless approach to mass detection in digital mammograms based on support vector machines. *Physics in Medicine and Biology*, 49:961–975, 2004.

[203] S. Nedjati-Gilani. Automated segmentation of multiple sclerosis lesions. Master's thesis, University College London, 2004.

[204] C. Dubreu. Active contour models and support vector machines: SVM snakes. Master's thesis, University College London, 2005.

[205] T. Shepherd and D.C. Alexander. A support vector machine for 3D texture-based classification of bone in CT images. In *Proceedings, Medical Image Understanding and Analysis*, pages 221–225, 2006.

[206] J. Mourão-Miranda, A. L.W. Bokde, C. Born, H. Hampel, and M. Stetter. Classifying brain states and determining the discriminating activation patterns: Support vector machine on functional MRI data. *Neuroimage*, 28:980–995, 2005.

[207] K Crammer, J. Kandola, and Y.Singer. Online classification on a budget. *Advances in Neural Information Processing Systems*, 16, 2004.

[208] J. Weston, A. Bordes, and L. Bottou. Online (and offline) on an even tighter budget. *Artificial Intelligence and Statistics*, pages 413–420, 2005.

[209] O. Dekel, S. Shalev-Schwartz, and Y. Singer. The forgetron: A kernel-based perceptron on a fixed budget. *Advances in Neural Information Processing Systems*, 18:259–266, 2006.

[210] G. Cavallanti, N. Cesa-Bianchi, and C. Gentile. Tracking the best hyperplane with a simple budget perceptron. *Machine Learning*, 69:143–167, 2007.

[211] N. Murata, K.-R. Muller, A. Ziehe, and S.-I. Amari. Adaptive on-line learning algorithms in changing environments. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information processing Systems*, page 599. Cambridge MA:MIT Press, 1997.

[212] N. Murata, M. Kawanabe, A. Ziehe, K.-R. Muller, and S.-I. Amari. On-line learning in changing environments with application sin supervised and unsupervised learning. *Neural Networks*, 15:743–760, 2002.

[213] A. Bordes, S. Ertekin, J. Weston, and L. Bottou. Fast kernel classifiers with online and active learning. *Journal of Machine Learning Research*, 6:1579–1619, 2005.

[214] I. Tsang, J. Kwok, and P.-M. Cheung. Core vector machines: fast SVM training on very large data sets. *Journal of Machine Learning Research*, 6:363–392, 2005.

[215] T. Joachims. Making large scale SVMs practical. In B. Scholkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 169–184. Cambridge MA:MIT Press, 1999.

[216] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[217] P. Laskov, C. Gehl, S. Kruger, and K-R. Muller. Incremental support vector learning: Analysis, implementation and applications. *Journal of Machine Learning Research*, 7:1909–1936, 2006.

[218] O. Barzilay and V.L. Brailovsky. On domain knowledge and feature selection using a support vector machine. *Pattern Recognition Letters*, 20:457–484, 1999.

[219] C. E. Metz. Evaluation of medical images. In A. E. Todd-Pokropek and M. A. Viergever, editors, *Medical Images: Formation, Handling and Evaluation*, pages 277–300. Springer Verlag, 1992.

[220] D. C. Alexander. *Statistical Modelling of Colour Data and Model Selection for Region Tracking*. PhD thesis, University College London, 1997.

[221] J. A. Swets. ROC analysis applied to the analysis of medical imaging techniques. *Investigative Radiology*, 14:109–121, 1979.

[222] Y. Zhan and D. Shen. Design efficient support vector machine for fast classification. *Pattern Recognition*, 38:157–161, 2005.

[223] T. Kuusela. Stochastic heart rate model can reveal pathologic cardiac dynamics. *Physical Review E*, 69, 2004.

[224] G. R. Jafari, S. M. Fazeli, F. Ghasemi, S. M. V. Allaei, M. R. R. Tabar, A. I. Zad, and G. Kavei. Stochastic analysis and regeneration of rough surfaces. *Physical Review Letters*, 91:226101–, 2003.

[225] M. Moradi, P. Mousavi, R. Siemens, E. Sauerbrei, A. Boag, and P. Abolmaesumi. Prostate cancer probability maps based on ultrasound RF time series and SVM classifiers. In *Proceedings, Medical Image Computing and Computer-Assisted Intervention*, pages 76–84, 2008.

[226] K. Itô. On stochastic differenial equations in a differentiable manifold. *Nagoya Mathematical Journal*, 1, 1950.

[227] K. Itô. On a formula concerning stochastic differentials. *Nagoya Mathematical Journal*, 3:55–65, 1951.

[228] H. Risken. Methods of solution and applications. In *The Fokker Planck Equation. Springer Series in Synergetics, 2nd ed.*, volume 18. Springer-Verlag, Berlin, 1989.

[229] R. Friedrich and J. Peinke. Description of a turbulent cascade by a Fokker-Planck equation. *Physical Review Letters*, 78:863, 1997.

[230] S. Kriso, R. Friedrich, J. Peinke, and P. Wagner. Reconstruction of dynamical equations for traffic flow. *Physics Letters A*, 299:287, 2002.

[231] P. Sura and S. T. Gille. Interpreting wind-driven southern ocean variability in a stochastic framework. *Journal of Marine Research*, 61:313–334, 2003.

[232] T. Kuusela, T. Shepherd, and J. Hietarinta. A stochastic model for heart rate fluctuations. *Physical Review E*, 67, 2003.

[233] R. Friedrich, S. Siegert, J. Peinke, S. Luck, M. Siefert, M. Lindemann, J. Raethjen, G. Deuschl, and G. Pfister. Extracting model equations from experimental data. *Physics Letters A*, 271:217–222, 2000.

[234] J. Timmer. Parameter estimation in nonlinear stochastic differential equations. *Chaos, Solitons and Fractals*, 11:2571–2578, 2000.

[235] A. S. Hurn, K. A. Lindsay, and V. L. Martin. On the efficacy of simulated maximum likelihood for estimating the parameters of stochastic differential equations. *Journal of Time Series Analysis*, 24:45–63, 2003.

[236] D. Kleinhans and R. Friedrich. Maximum likelihood estimation of drift and diffusion functions, 2007.

[237] C. Gourieroux, A. Monfort, and E. C. Renault. Indirect inference. *Journal of Applied Econometrics*, 8:85–118, 1993.

[238] A. R. Gallant and G. Tauchen. Which moments to match? *Econometric Theory*, 12:657–81, 1996.

[239] P. E. Kloeden and E. Platen. *Numerical Solution of Stochastic Differential Equations*. Springer, Berlin, 1992.

[240] A. M. Stuart, J. Voss, and P. Wiberg. Conditional path sampling of SDEs and the Langevin MCMC method. *Communications in Mathematical Sciences*, 2:685–697, 2004.

[241] A. Apte, M. Hairer, A. M. Stuart, and J. Voss. Sampling the posterior: An approach to non-Gaussian data assimilation. *Physica D: Nonlinear Phenomena*, 230:50–64, 2007.

[242] C. Archambeau, D. Cornford, M. Opper, and J. Shawe-Taylor. Gaussian process approximations of stochastic differential equations. *Proceedings, JMLR Workshop on Gaussian Processes in Practice.*, 1:1–16, 2007.

[243] H. Haken. Introduction and advanced topics. In *Springer Series in Synergetics*. Springer-Verlag, Berlin, 2004.

[244] C. W. Gardiner. *Handbook of stochastic methods for physics, chemistry and the natural sciences*, chapter 13. Springer Series in Synergetics, 3rd ed. Springer-Verlag, Berlin, 2004.

[245] R. Friedrich, J. Peinke, and C. Renner. How to quantify deterministic and random influences on the statistics of the foreign exchange market. *Physical Review Letters*, 84:5224–, 2000.

[246] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[247] C. K. I. Williams and C. E. Rasmussen. Gaussian processes for regression. *Advances in Neural Information Processing Systems*, 8:514–520, 1996.

[248] R. M. Neal. Monte Carlo implementation of Gaussian process models for bayesian regression and classification. Technical Report 9702, University of Toronto, Deptartment of Statistics, 1997.

[249] S. Brahim-Belhouari and A. Bermak. Gaussian process for nonstationary time series prediction. *Computational Statistics and Data Analysis*, 47:705–712, 2004.

[250] K. V. Mardia and R. J. Marshall. Maximum likelihood estimation for models of residual covariance in spatial regression. *Biometrika*, 71:135–146, 1984.

[251] P. K. Kitanidis. Statistical estimation of polynomial generalized covariance functions in hydrologic applications. *Water Resources Research*, 19:909–921, 1983.

[252] J. J. Warnes. Problems with likelihood estimation of covariance functions of spatial Gaussian processes. *Biometrika*, 73:640–642, 1987.

[253] A. Schwaighofer, V. Tresp, and K. Yu. Learning Gaussian processes from multiple tasks. In *Proceedings, ACM International Conference*, pages 1017–1024, 2005.

[254] J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models. In *Advances in Neural Information Processing Systems*, volume 18, pages 1441–1448. MIT Press, 2006. Proc. NIPS'05.

[255] R. Urtasun, D.J. Fleet, and P. Fua. 3D people tracking with Gaussian process dynamical models. In *Proceedings, IEEE Conference on Computer Vision and Pattern Recognition*, pages 238–245, 2006.

[256] D. Cremers. Dynamical statistical shape priors for level set-based tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28, 2006.

[257] A Goldberger. Is the normal heartbeat chaotic or homeostatic? *News in Physiological Sciences*, 6:87–9, 1991.

[258] U Zwiener, D Hoyer, R Bauer, B Lüthke, B Walter, K Schmidt, S Hallmeyer, B Kratzsch, and M Eiselt. Deterministic-chaotic and periodic properties of heart rate and arterial pressure fluctuations and their mediation in piglets. *Cardiovascular Research*, 31:455–465, 1996.

[259] J. B. Bassingthwaighte and G. M. Raymond. Evaluation of the dispersional analysis method for fractal time series. *Annals of Biomedical Engineering*, 23:491–505, 1995.

[260] B Hao. Symbolic dynamics and characterization of complexity. *Physica D*, 51:161–176, 1991.

[261] I. A. Rezek and S. J. Roberts. Stochastic complexity measures for physiological signal analysis. *IEEE Transactions on Biomedical Engineering*, 45:1186–1191, 1998.

[262] J. A. Palazzolo, F. G. Estafanous, and P. A. Murray. Entropy measures of heart rate variation in conscious dogs. *American Journal of Physiology*, 274:1099–1105, 1998.

[263] S Pincus. Approximate entropy (ApEn) as a complexity measure. *Chaos*, 5:110–117, 1995.

[264] D. Drasdo and S. Höhme. A single-cell-based model of tumor growth in vitro: monolayers and spheroids. *Physical Biology*, 2:133–147, 2005.

[265] K. Khairy, E. Reynaud, and E. Stelzer. Detection of deformable objects in 3D images using Markov-Chain Monte Carlo and spherical harmonics. In *Proceedings, Medical Image Computing and Computer-Assisted Intervention*, pages 1075–1082, 2008.

[266] M. Ferrant, S. K. Warfield, A. Nabavi, F. A. Jolesz, and R. Kikinis. Registration of 3D intraoperative MR images of the brain using a finite element biomechanical model. *IEEE Transactions on Medical Imaging*, 20:1384–1397, 2001.

[267] K. V. Mardia. *Directional Statistics*. Wiley, 2000.

[268] J. E. Bresenham. Algorithm for computer control of a digital plotter. *IBM Journal of Research and Development*, 1965.

[269] O. Friman and C-F. Westin. Uncertainty in white matter fiber tractography. In *Proceedings, Medical Image Computing and Computer-Assisted Intervention*, pages 107–114, 2005.

[270] S. Jbabdi, M. W. Woolrich, J. L. R. Andersson, and T. E. J. Behrens. A bayesian framework for global tractography. *Neuroimage*, 37:116–129, 2007.

[271] F. A. Escobedo and J. J. de Pablo. Extended continuum configurational bias Monte Carlo methods for simulation of flexible molecules. *Journal of Chemical Physics*, 102:2636–2652, 1995.

[272] R. Akbani, S. Kwek, and N. Japkowicz. Applying support vector machines to imbalanced datasets. In *Proceedings, European Conference on Machine Learning*, pages 39–50, 2004.

[273] D. Kendall. Shape manifolds, Procrustean metrics and complex projective spaces. *Bulletin of the London Mathematical Society*, 16:81–121, 1984.

[274] D. Kleinhans, R. Friedrich, and J. Peinke. An iterative procedure for the estimation of drift and diffusion coefficients of Langevin processes. *Physics Letters A*, 346:42–46, 2005.

[275] D. J. C. MacKay. Introduction to Gaussian processes. In C. M. Bishop, editor, *Neural Networks and Machine Learning*, pages 84–92. Springer Verlag, 1998.

[276] C. F. Lucchinetti, W. Bruck, M. Rodriguez, and H Lassmann. Distinct patterns of multiple sclerosis pathology indicates heterogeneity on pathogenesis. *Brain Pathology*, 6:259–274, 1996.

[277] R. M. May. Biological populations with nonoverlapping generations: Stable points, stable cycles, and chaos. *Science*, 186:645–647, 1974.

[278] J. Wang and R. Engelmann. Segmentation of pulmonary nodules in three-dimensional ct images by use of a spiral-scanning technique. *Medical Physics*, 34:4678–4689, 2007.

[279] T. F. Cootes, G. J. Edwards, and C. Taylor. Active appearance models. In *Proceedings, European Conference on Computer Vision*, pages 484 – 498, 1998.